UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

EMOSTATUS – ESTIMATING THE EMOTIONAL STATE OF PEOPLE USING ELECTRODERMAL ACTIVITY

João Gonçalo Marques Oliveira

Mestrado em Engenharia Informática
Especialização em Engenharia de Software

Dissertação orientada por:
Prof. Doutor Manuel João Caneira Monteiro da Fonseca

2019

Resumo

O reconhecimento de emoções é uma área onde a investigação é cada vez mais importante, devido ao número grande de áreas em se pode tirar partido das emoções. Desde sessões de terapia até divertimento com a possibilidade de alterar os eventos de um videojogo dependendo das emoções da pessoa, existem muitas razões para estudar nesta área. No entanto, os trabalhos feitos até agora têm alguns problemas. Primeiro, uma grande parte dos investigadores identifica um número reduzido de emoções, perdendo muito detalhe emocional. Outro problema é que, embora existam artigos que demonstram diferenças entre as respostas fisiológicas a estímulos entre pessoas do sexo masculino e feminino, poucos investigadores investigaram isto e fazem os seus algoritmos com os dois sexos combinados. Por último, os investigadores fazem os seus algoritmos para optimizar a eficácia sem se preocuparem com a eficiência, resultando na utilização de vários métodos complexos que não se podem usar para efectuar estimação em tempo real.

Com este trabalho visamos desenvolver um algoritmo que consiga estimar qualquer emoção em tempo real, e verificar se a separação dos sexos masculino e feminino influência de forma positiva esta estimação. Para facilitar a criação do algoritmo e de forma a garantir que o algoritmo executa o mais rápido possível, criamos também uma framework para o desenvolvimento de algoritmos de reconhecimento de emoções.

Para tal, estudámos os métodos que os investigadores utilizam nesta área. Rapidamente se verificou que a maioria dos trabalhos efectuados tendem a seguir os mesmos passos básicos. Primeiro, o sinal, retirado de um dataset é pré-processado com métodos para retirar ruído ou de segmentação de sinal. Depois as características são extraídas dos sinais. Um grande número de características pode ser extraído, pelo que alguns investigadores decidem usar algum procedimento para reduzir o seu número. Um método utilizado é a avaliação da correlação para identificar se existem características que não contribuem para melhorar o resultado. Outro método é a utilização de algoritmos de selecção de características como o PCA (*Principal Component Analysis*) que identifica as melhores características e reduz a dimensionalidade. De seguida, as características obtidas são utilizadas para treinar um modelo de aprendizagem automática. Modelos comuns incluém SVM, *Random Forest*, Árvores de Decisão, entre outros.

Para efectuar a estimação de emoções em tempo real, escolhemos a Atividade Eletrodérmica, um dos sinais mais utilizados na área. No entanto verificamos outros sinais, como o Electrocardiograma (ECG) e a Pupilometria. Verificámos que embora estes sinais sejam também muito usados, especialmente o ECG, estes têm factores negativos que tornam a sua utilização neste trabalho não útil.

Estudámos também os modelos possíveis para representação de emoções, de modo a obter um número mais rico de emoções. O modelo identificado como ideal foi o *Circumplex Model of Affect*, que mapeia valores de valência e excitação num espaço 2D contínuo e representa o espaço emocional completo. Decidimos então estimar valores de valência e excitação separadamente para obter um ponto no *Circumplex Model of Affect*.

Depois de examinar os trabalhos dos investigadores, escolhemos os métodos para usar no nosso trabalho. Com base no facto que para conseguir estimar valores em tempo real precisamos de algoritmos rápidos, escolhemos utilizar algoritmos comuns que obtêm bons resultados e que são eficientes. Em termos de pré-processamento, segmentamos o sinal em janelas de cinco segundos com 50% de sobreposição, obtendo assim uma estimação a cada 2.5 segundos de sinal, possibilitando assim ver o progresso do estado emocional ao longo do tempo. Cada segmento de cinco segundos é depois submetido à eliminação de ruído através de uma *wavelet Daubechies*. De seguida, características de estatística no domínio do tempo são extraídas de cada segmento. As características utilizadas são o máximo, o mínimo, a diferença entre o máximo e mínimo, o desvio padrão, a assimetria e a curtose. Estas características são então utilizadas para treinar árvores de decisão, pois foi o modelo de aprendizagem automática onde obtivemos os melhores resultados.

Depois de determinar quais os métodos a utilizar, construímos a nossa framework. A framework é constituida por vários blocos representativos de cada passo básico do processo de estimação que foram identificados durante a leitura de artigos. Estes blocos genéricos estão construídos de maneira a serem facilmente extensíveis para um grande número de métodos diferentes, não limitados ao nosso algoritmo, podendo ser usados com outros algoritmos e sinais fisiológicos. Os blocos executam paralelamente e comunicam entre si de maneira rápida, possibilitando estimações em tempo real.

O nosso algoritmo para identificar o estado emocional foi implementado na framework e várias experiências foram feitas com dois datasets diferentes: o '*A Dataset for Affect, Personality and Mood Research on Individuals and Groups*' (AMIGOS) e o '*A Database for Emotion Analysis using Physiological Signals*' (DEAP). Os resultados mostram que o erro de estimação no dataset AMIGOS é muito baixo, obtendo um erro médio de 0.011 para valência e 0.009 para excitação, na estimação com ambos os sexos, para a escala de [-0.5,0.5], e com o dataset DEAP obtém resultados um pouco piores, com erro médio de 0.055 para valência e 0.056 para excitação. Quanto aos resultados dos sexos separados, no dataset AMIGOS o sexo masculino atingiu resultados ligeiramente melhores (erro médio de 0.009 e 0.007 para valência e excitação, respectivamente) e o sexo feminino obteve resultados ligeiramente melhores (erro médio de 0.012 para valência e 0.018 para excitação, respectivamente). No dataset DEAP, ambos os sexos obteram resultados melhores que o estimador com ambos. O sexo masculino obteve um erro médio de 0.047 para valência e 0.044 para excitação, enquanto que o sexo feminino obteve erro médio de 0.049 para valência e 0.046 para excitação. Extrapolando estes resultados para classificação dos quadrantes do *Circumplex Model of Affect*, o algoritmo obteve precisões de 96%, 97% e 95% para as estimações com ambos os sexos, masculino e feminino, respectivamente no dataset AMIGOS, enquanto que o DEAP conseguiu apenas 82% para am-

bos os sexos e 85% para masculino e feminino. Os resultados obtidos são melhores que muitos dos trabalhos já efectuados na área, especialmente os do AMIGOS.

Destes resultados pode-se concluir que, embora os resultados no DEAP não sejam tão bons como os dos AMIGOS, continuam a ser bons, pelo que o algoritmo pode atingir bons resultados em diferentes datasets. No entanto, os resultados podem ser potencialmente melhorados no futuro através da revisão das características utilizadas. Também se pode especular que os resultados relativamente piores do sexo feminino no dataset AMIGOS podem estar relacionados com o pequeno número de elementos de treino, pois o número de dados para o sexo feminino é metade do número de dados do sexo masculino neste dataset

Finalmente, conseguimos um tempo de execução muito baixo, na ordem dos 10ms, pelo que o algoritmo pode ser utilizado para fazer estimação em tempo real.

Palavras-chave: Reconhecimento de emoções, Tempo real, Excitação, Valência, Atividade eletrodérmica

Abstract

Emotional recognition is an area with growing importance, with applications in areas such as medicine, advertising and even software design. Electrodermal Activity is one of the physiological signals most used to perform emotional recognition. There are many ways researchers use this signal to predict emotions, but generally they use a small set of emotions, are not concerned with the speed of the algorithm, and very few look into the differences between men and women. As such, this work intends to develop an algorithm that can predict any emotion in real-time and to determine if separating data from men and women improves the results. To do so, we studied the current methods for emotion recognition and chose the ones that best fit our purposes in terms of speed and accuracy. We also identified the common general steps that most researchers use for emotion recognition. With algorithmic speed in mind, and with the knowledge obtained from the research, we built a general purpose emotional recognition framework which uses small blocks that communicate amongst each other and execute in paralell, removing any possible delay in the estimation thus allowing real-time estimation. We implemented our algorithm using this framework. Experimental evaluation showed that our algorithm achieves estimations with very small errors in the AMIGOS dataset and an accuracy for the estimation of quadrants of 96% for both genders, 97% for males and 94% for females. For the DEAP dataset, values of 82% for both genders and 85% for males and females were achieved. When compared with existing works, our algorithm presents better results, both for the estimation of valence and arousal and for the estimation of the quadrants. Finally, our algorithm performs its estimations in under 10ms, therefore it can be used for real-time experiments.

Keywords: Emotion recognition, Real time, Arousal, Valence, Electrodermal activity

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this introductory chapter we give a brief overview of our work. First, we present the motivation that brought about this work. Afterwards we state our goals for the work, summarize the main contributions of this work and enumerate some of the results obtained. Finally, we describe how the rest of the document is structured.

1.1 Motivation

Emotion recognition has become an important research area. Being able to pinpoint accurately what emotion a person is experiencing has been shown to be beneficial in many different scenarios, such as improving the effectiveness of therapy sessions, gathering feedback on how users perceive certain media such as advertisements or software interfaces and even dynamically adjusting content in a videogame depending on how a player feels. Therefore, it is useful to have computer systems that can automatically detect emotions being felt by people.

However, research done tends to have a few problems. Researchers generally identify only sets of five or less different emotions, limiting the amount of emotions that can be estimated. The second problem is that even though research has shown males have different physiological reactions than females for the same emotion, many works combine the two without analysing if their algorithm loses accuracy by doing so. The third problem is that researchers ignore the time taken by algorithms to compute the emotion based on the subject's physiological signals, which is fundamental to obtain real-time results.

In addition, the development of emotional recognition algorithms is complex. To make it easier, many researchers plan their system structure before implementing it. However, these systems are specific to the researcher's solution and usually not public. Therefore, it is useful to have a general purpose framework that can support the development of different algorithms.

This work was conducted at LASIGE, a research unit at the the Department of Informatics, Faculty of Sciences, University of Lisbon, in the context of the project Awareness While Experiencing and Surfing On Movies through Emotions (AWESOME), supported by the Fundação para a Ciência e Tecnologia (FCT) under LASIGE Strategic Project - UID/CEC/00408/2019, and under project AWESOME - PTDC/CCI- INF/29234/201.

1.2 Goals

The main goal of this work is to develop an algorithm that can determine a subject's emotional state after being exposed to a stimuli (e.g. image, video, sound, etc.) from their physiological signals. In particular, we use Electodermal Activity (EDA), also known as Galvatic Skin Response to estimate the emotional state for a rich set of emotions and in real-time. Additionally, we want to develop a software framework to make the creation and testing of emotion recognition algorithms easier and faster. To fullfill this, the following secondary goals must be accomplished:

- Research current methods for recognizing emotions from physiologicals signal and determine which are most useful for our goal;
- Design and implement the framework;
- Implement the algorithm for emotion estimation according to the methods identified in the research;
- Build experiments to evaluate and validate the algorithm, and determine if separating male and female subjects improves results.

1.3 Developed Solution

As mentioned in the previous section, our solution is split into two parts: a framework that supports the development of algorithms for emotional recognition, and an algorithmic process that identifies the emotional state of subjects based on their physiological signals.

The developed framework is composed of several blocks, each representing an independent part of the emotion recognition process. These blocks are Signal Feed, Preprocessor, Feature Extractor, Estimator and Results. The blocks can be connected between each other, making it simple to create emotional recognition algorithms and experiments.

The algorithm for emotional recognition was built using this framework, based on the results of the research on the current state of the art. Specifically, we start by gathering the signal from a dataset, then we split the signal into five second segments with 2.5 seconds of overlap and denoise each segment individually with the Daubechies4 wavelet. Then we extract six time-based statistical features and finally use those features to train and estimate with decision trees.

In order to obtain a richer set of emotions, instead of identifying individual emotions, we estimate values for Valence and Arousal, which are described in a 2D space. Thus, with an estimated value for Valence and another for Arousal, any emotion can be represented, providing us with a richer emotional space.

1.4 Main Contributions

This work has two main contributions. The first is the creation of an algorithm for estimating people's emotional state, that is a new combination of methods already used in previous works, but selected and combined to achieve good results and performance in real-time. The estimated values, on average, have very low deviation from the expected values, allowing the algorithm to obtain high accuracies. Some of the gender-specific estimators obtained slightly better results while others achieved slightly worse results, but the difference was not significant.

The second main contribution is the new framework for emotional recognition which allows building algorithms in a simple and fast way, as well as to perform experiments easily. The framework helps making more efficient algorithms, since it has parallelized components that quickly trade data through communication channels. With the help of this framework the algorithm executes efficiently, with the first prediction taking below 100ms and every subsequent predictions taking less than 10ms.

1.5 Document Structure

This document is composed by six chapters.

In the first chapter, **Introduction**, we present the motivation and goals of the work, as well as a summary of the solution and the main contributions.

In the second chapter, **Background and State of the Art**, we analyze the research that has been done in the area.

In the third chapter, **Algorithm for Estimation of the Emotional State**, we take our findings from the second chapter and choose the algorithms we will use in our work

In the fourth chapter, **Emotional Recognition Framework**, we detail the framework we built to aid in constructing our algorithm.

In the fifth chapter, **Evaluation**, we present our results.

Finally, in the sixth chapter, **Conclusions and Future Work**, we summarize and take conclusions about the work and its future.

Chapter 2

Background and State of the Art

As mentioned in Chapter 1, the first step to determine the algorithm for emotional recognition was to examine the current state of the art in emotional recognition from physiological signals. In this chapter we describe the current state of the art, starting by giving some background about the way emotions are represented. Then we discuss the physiological signal we chose for this work, Electrodermal Activity, and list methods to pre-process the signal, the kind of features typically used, how features are selected and which estimators are most used, and then present some works that use EDA. Afterwards we give a brief overlook of other signals that could have been chosen. Finally we summarize and discuss our findings.

2.1 Representation of Emotions

In order to estimate emotions, researchers have to describe emotions using models. There are two common approaches to achieve this.

The first approach is to choose a set of distinct emotions. Researchers estimate for limited sets of two[34][4], three[12], four[15][17] or five[13][21][32] emotions. Although there are some researchers who use more emotions, there are two problems. The first problem is that more emotions can make estimation less accurate. The second problem is that the datasets used do not have enough data for the less common emotions, and thus is not possible to train a balanced model[32].

The second approach is to define emotions according to the Circumplex Model of Affect[25]. This model maps emotions in a 2D space, where the vertical axis represents Arousal and the horizontal axis represents Valence. Arousal represents how intense an emotion is and Valence says whether the emotion is negative or positive. The placing of some emotions in this model can be seen in Figure 2.1.

Researchers estimate emotions based on the Circumplex Model of Affect in two different ways. One option is to have Low and High for both Arousal and Valence as classes, for a total of four classes, and then classify which quadrant the emotion is in. This can be done with Arousal and Valence combined[13] or separately[13][30][35][31]. When classified separately, the results may be joined to obtain the corresponding quadrant[29]. The other option is to estimate values of Arousal and Valence to obtain specific positions in the 2D space[23]. This

approach provides a richer description of the emotional state and will be the one used in our work.

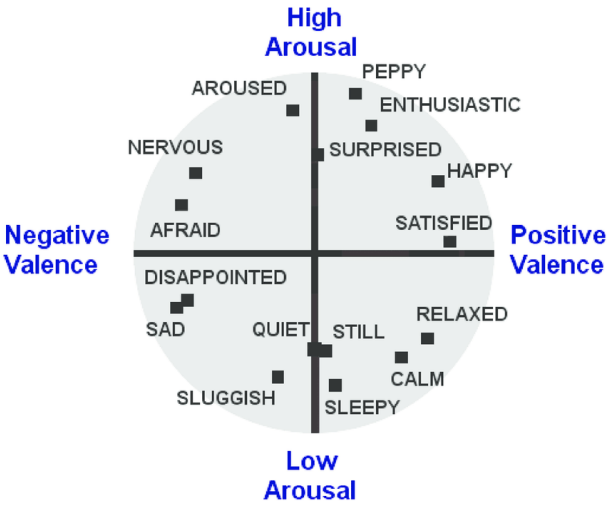

Figure 2.1: Visual representation of the Circumplex Model of Affect. Image adapted from [3].

2.2 Electrodermal Activity

Electrodermal Activity, or Galvanic Skin Response, refers to the change in sweat gland activity. Research has shown that this change reflects the intensity of the emotional state, making EDA a good measure of Arousal [26]. Because of this, EDA has become a common physiological signal to use for emotional estimation.

To estimate emotions, researchers use a series of steps which can be generalized into four groups. First, the signal, obtained either through a dataset or by an experiment, is pre-processed, which typically involves transforming the signal to remove discrepancies induced by factors other than the stimulus. Afterwards its features are extracted. Some researchers then opt in using methods to perform feature selection. Finally, the features are given to a classifier or estimator, for training or classification/estimation. The next sections present several techniques used in each of these steps.

2.2.1 Pre-processing

As mentioned before, the goal of pre-processing is to ensure the signal is prepared for the extraction of its features. A signal may, for example, have noise or interference that needs to be removed, or it may be too long and need segmentation for better feature extraction.

A common method to deal with interference is noise removal. To remove noise the common approach is to use filters. Low-pass filters attenuate the signal when frequencies above a given frequency, while high-pass filters attenuate the signal under a given frequency. Band-pass filters act as both a low-pass and high-pass filter. EDA signals do not overlap with other signals in the band 0.08-0.2Hz[21]. Researchers however use different cut-off values. Wang et. al use a

low-pass filter 3Hz cut-off[31], Zangróniz et al. use a low-pass filter with 1.5Hz cut-off[34], Yang et al. use a low-pass filter with 60Hz cut-off[33] and Das et al. use an elliptical filter, a kind of band-pass filter, with 10Hz cut-off[12]. Notch filters, which are a type of band-pass, are frequently used to remove inaccuracies caused by the hardware collecting the signal. Some researchers opt to use a powerline notch filter to remove powerline interference[15]. Powerline interference has a frequency of 50Hz to 60Hz[15], so researchers use cut-off frequencies like 48-52Hz[29] and 50Hz[13]. Another common way to remove noise is through the use of wavelet smoothing [21][35], which also functions as a band-pass filter[21]. The most common wavelets to smoothen EDA signals are Coiflet and Daubechies.

EDA signals are composed by two different components: skin conductance response (SCR) and skin conductance level (SCL). SCL can be seen as a "baseline" for skin conductance, which varies between different people but very little for the same person depending on their physiological state, while the SCR reflects most of the change in the signal. One approach, used by Zangróniz, et al., was to separate these two components using a deconvolution operation so the SCR can be analyzed by itself[34]. The researchers note that this method achieves better performance at the cost of more intensive signal processing.

Depending on the dataset or the equipment, the signal may be collected at higher frequencies than needed. On higher frequencies, constant small flunctuations of the signal are considered noise, so some researchers choose to downsample their data. A technical report by researchers from the University of Birmingham, UK, suggests that because EDA signal does not change very quickly, sample rates as low as 70Hz are typical, though it is recommended a sample rate of 200Hz - 400Hz for separation of the SCR and SCL[8]. Many researchers use frequencies in the range of 128Hz - 256Hz[32][4], though some use higher[13][15] and some use frequencies as low as 10Hz[34][29].

As with other kinds of signals, a common approach for analysis of data that is longer than a few seconds is to split the data into smaller segments, also called epochs. Typically these segments have some overlap with the previous and the next segments. Segment length and overlap varies between works. For segment length, researchers generally use lengths between five and twenty seconds. For overlap, 50% overlap is common, but Anderson, et al. claimed to achieve better results with twenty second segments and 80% overlap[4]. This approach is used to provide analysis of smaller segments of the signal, from which more meaningful information can be extracted.

2.2.2 Feature Extraction

Features are extracted from the pre-processed signal and then used to train a machine learning model or to get an estimation from it. There are many kinds of features and researchers select many of them for their procedures, including:

- **Time-based statistical features** - these features are simple to calculate and are used by many researchers[34][21][4][21][35][27]. Examples of time-based statistical features include maximum, minimum, dynamic range, mean, standard deviation, kurtosis and skew-

ness. Some researchers extract these features from the first and second derivatives of the signal[34][32].

- **Frequency-based statistical features** - Frequency-features can be extracted from the signal by first applying methods such as Fast Fourier Transform or Power Spectral Density to obtain the frequency-domain representation of the signal and then extracting features like spectral power[31]. Researchers also commonly use the same kind of features as the time-based features to extract further information from the frequency-domain signal[28].

- **Entropy-based features** - Entropy-based features, which analyze the irregularity of the signal, are also very used[30][4][31][15]. Many kinds of entropy features are used, like Sample Entropy, Approximate Entropy, Low Energy Entropy, Shannon Entropy, etc.

- **Other features** - Many works use various types of features other than the previous three, but they are not as frequently used as these. Some methods, like Poincaré Plots, are often used to plot variations in heart rate extracted from ECG signals, but are not as commonly used for EDA signals, though some researchers use them to plot variation in EDA signals[15]. Other less common methods include Recurrence Quantification Analysis, Lyapunov exponents, Detrended Flunctuation Analysis, Kolmogorov, Triangle Space Mapping[15], Wavelet Transform[14], Hjorth Features and features related to Mel-frequency cepstrum[28].

2.2.3 Feature Selection

During the Feature Extraction process, researchers often extract several features. However, with the increase in the number of features comes an increase in the complexity of training the classifier or estimator. To reduce the dimensionality, a number of feature selection methods can be applied. One method is to calculate statistical significance between different classes and then analyze the results to determine the most important features[34][35]. However, for a big number of features and many possible estimation labels, this process can be very complex. One possible solution is to apply methods like Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA). PCA automatically optimizes the covariance and LDA maximizes linear class differentiation in a low-dimensional space[13].

2.2.4 Classifiers and Estimators

As mentioned in the Emotion Estimation section, most of the research estimates emotions with classifiers, which choose a result from a limited number of possibilities. For regression, researchers use estimators based on the same principles as the classifiers.

Support Vector Machines (SVM)[7] define an hyperplane that separates two classes in a way that minimizes incorrect predictions. Because of this its training process can take a long time, but predictions are significantly quicker.

K-Nearest Neighbors (KNN)[6] are a simple classifier that calculates the distance between every other point to calculate its nearest neighbors. In classification problems, these neighbors are used to determine the class by checking the number of occurances for each class among the neighbors. In regression problems it is similar, but instead of calculating the class it averages the values corresponding to the neighbors to determine the estimation. While simple, with a lot of data it tends to be slower than the alternatives, as it must check against every data point every time it estimates.

Decision Trees[6] are classifiers that build a set of splitting nodes. Each node checks if the value of a feature is higher or lower than a threshold and chooses the next node to check based on the result. Since every node is a simple comparison, it runs very quickly.

Random Forests[20] are classifiers that internally use a number of Decision Trees. Each tree is randomly assigned a subset of features to get an estimation, and the final estimation is the combination of the results. This method overcomes some common problems that Decision Trees have with overfitting and usually improves results[20].

Naive Bayes[22] is a probabilistic classifier where each feature contributes a certain probability that one data point is from a class. It is still used, though research has shown that most recent classifiers outperform it on average[9].

Other less used classifiers include Matching Pursuit, Quadratic classifier, Fisher classifier, various methods based on neural networks such as Multi-Layer Perceptron and Deep Convolutional Neural Networks, and boosting-based techniques like AdaBoost and XGBoost.

2.3 Main EDA works

In the previous section we presented various methods that researchers use for each of the steps of the algorithm. In this section, we present a few works that obtained the best results for the overall algorithm for emotion recognition.

Zangróniz, et al. researched the possiblity of using GSR to determine if participants are feeling calm or distressed when presented with image stimuli[34]. The raw signal was preprocessed by applying a low-pass FIR filter with 1.5Hz cut-off to decrease noise. The signal was then decomposed with a deconvolution operation, separating the SCL and SCR. The features used were time-based statistical features computed over the SCR component, which were then selected by applying an ANOVA test and discarding those that were not statistically significant. Using a Decision Tree as classifier, they obtained the overall classification accuracy of 89%.

Das, et al. researched the use of GSR and ECG separately and combined to classify three emotions (happy, sad, neutral)[12] on signals obtained from visualization of 154-second long videos. Elliptical filters with frequency cut-off at 10Hz was used for the EDA signal. For the ECG signal, wavelet decomposition at level 12 was applied with mother wavelet db6. The ECG signal was further smoothened using a Savitsky-Golay filter. An FIR filter was applied to both signals to remove high frequency interference. The features used were Welch's Power Spectral Density at the range of 0 to 20Hz and six statistical features: mean, median, mode, variance, kurtosis and skewness. Though they classified three emotions, they opted to run their

experiments in sets of two. Their results revealed that the classification with the EDA signal was better on all feature sets and classifiers than the ECG signal. When classifying between the two extreme emotions, happy and sad, the statistical features obtained 100% accuracy with the EDA signal. When classifying between Happy and Neutral statistical features were only slightly better than PSD, with the best result of 83.13% being obtained with the SVM classifier. For the Sad-Neutral classification, KNN with PSD features obtained 100%, while the best result for the statistical features was 81.61%. When the PSD and statistical features were both used in classification, the results for Happy-Sad classification fell to approximately 78% with SVM and KNN, while Happy-Neutral improved to 90.58% with Naive Bayes, though SVM fell to 62.58%, twenty percentage points lower than the statistical features alone. Sad-Neutral obtained 100% with SVM.

Goshvarpour, et al. chose to classify four classes (Happiness, Peacefulness, Sadness and Fear) on data obtained from music stimuli. First a digital notch filter was used to remove powerline interference. Then the signal was split into 10-second segments with 50% overlap. Then they extracted the features using Poincaré plot indices, Recurrence Quantification Analysis, Entropy, Lyapunov exponents, Embedding dimension, Detrended Flunctuaton Analysis, Kolmogorov and Triangle Phase space mapping. They obtained a lot of features, for which they chose to compare three different feature selection methods: Sequential floating Forward Selection, Sequential Forward Selection and Random Subset Feature Selection. Finally they also chose to compare between four different classifiers: Least Squares Support Vector Machine, KNN, Quadratic classifier and Fisher classifier.The best result on average was achieved with Fisher classifier and Random Subset Feature Selection, with 87.53%, though the other feature selected methods were not far behind, with 87.42% and 87.44%.

Zhang et al., differently from most other researchers, chose to estimate values of Valence and Arousal with regression[35]. They used 5 signals: EDA, Photoplethysmography, Skin Temperature, Respiration Rate and Pupil Diameter. The signals were first denoised with a moving average filter and wavelet smoothing and then segmented into 10 second segments. Finally, the signals were decomposed by 6 levels of Daubechies5 wavelet transform. During the decomposition, high-pass and low-pass filters were used separately to collect coefficients to use for the feature extraction. For every signal, 432 statistical features were collected from the original and decomposed signals as well as 6 energy-based features. To reduce the number of features, the authors employed ANOVA analysis. The best result for Arousal was 0.02323 Mean Square Error (MSE) and 0.7347 Correlation Coefficient (CC), and a 0.01485 MSE and 0.7902 CC for Valence with NuSVR. Results for other estimators can be seen in Table 2.1.

2.4 Other Physiological Signals

As seen in some of the works presented in the previous section, Electrodermal Activity is not the only signal used for emotion recognition. Many works attempt to use two or more signals either to compare the two or to use them together in order to extract the advantages of each signal. Despite the choice of only using EDA in this work, it is still useful to analyze some

Algorithm	Arousal		Valence	
	MSE	CC	MSE	CC
Decision Tree	0.04778	0.2395	0.03727	0.2335
SVR(linear kernel)	0.04649	0.4371	0.02729	0.5762
Linear Regression	0.04359	0.4465	0.02675	0.5812
Ridge Regression	0.04334	0.4474	0.02674	0.5813
MLP	0.05573	0.5471	0.03551	0.6103
SVR(poly kernel)	0.03099	0.6426	0.02593	0.6574
K-Nearest Neighbors	0.02722	0.6774	0.01910	0.7179
SVR(rbf kernel)	0.02676	0.7031	0.01767	0.7605
NuSVR	**0.02323**	**0.7347**	**0.01485**	**0.7902**

Table 2.1: Table of the results obtained by Zhang, et al. Image adapted from [35].

works that use other signals.

Electrocardiogram (ECG) is a measure of the electric activity of the heart. Heart Rate (HR) and Heart Rate Variability (HRV) can be extracted from it, which gives even more information from which features can be extracted.

Wang, et. al used ECG along with EDA and Electroencephallography (EEG) [31]. Spectral power and statistical features were extracted from ECG, including analysis of the differences between consecutive R peaks in the QRS complex, which denote heart depolarization during a heart beat. By combining the features from all three signals, the authors did binary classification for High Low and obtained 80.1% accuracy for Valence and 68.4% for Arousal.

Moharreri, et al. chose to represent the differences between successive R peaks in a Poincaré plot and extra several features from it[23]. In the end, they estimated values of Valence and Arousal using separate Decision Trees, obtaining an MSE of 0.0536 and 0.0393 for Valence and Arousal respectively. When they convert the results to classification, they obtained an accuracy of 95.71%.

Despite there are some works with good results like Moharreri, et al., the ECG signal has the disadvantage that it requires a long time to extract some of its meaningful features. The approach used by Moharreri, et al. used 5 minutes of signal in order to obtain enough data to fill the Poincaré plot.

Another physiological signal that could be used is Pupillometry, which measures pupil diameter.

Babiker, et al. had success in using slight differences in pupil diameter between positive and negative stimuli to achieve a 96.50% recognition rate in valence, in an experiment where audio stimulus was used[5].

However, pupil diameter is very influenced by luminosity, and as such, in an experiment with image or video stimuli, it would be hard to predict how much the increase or reduction of pupil diameter was due to the emotion and not the visuals[5][24][16]. Henderson, et al. did an experiment on emotional imagery, where subjects were presented with short scenarios (for example, "You jump up and block the volleyball at the net, saving the game") and told to imagine themselves in that scenario. One of the problems the authors had was that the scripts had to be very carefully chosen to minimize differences in brightness and contrast[16].

2.5 Discussion

In the previous sections we presented the methods and some of the works on emotional recognition. The main works that use EDA are summarized in Table 2.2.

Many of the works use a combination of simple and complex algorithms. Statistical-based features are simple and efficient to compute, are used in most works and generally achieve good results. However, works that combine statistical features with other kinds tend to achieved better results. In other cases, authors may calculate several features and then reduce their number by using Feature Selection algorithms. The problem of using many features and using these algorithms to reduce the dimensional space is that the calculations use a lot of computation time, which is undesireable for our work. The same problem with computation time occurs when using pre-processing methods like SCR/SCL separation. Also noteworthy that many works use the full length of the signal without segmenting it in epochs. This shows that most authors are typically not concerned with having a fast algorithm, but only in achieving the best results.

Another important aspect to highlight is the low number of emotions that are identified. Most researchers classify from two to five classes. Zhang, et al., however, estimated values of Valence and Arousal, using the full spectrum of the Circumplex Model of Affect. However, they did not combine the values of Valence and Arousal to take further conclusions on the accuracy of their solution.

Due to the wide range of methods used, it is difficult to determine which parts contribute most to get good results. Many works also use several signals, further raising complexity. Works using several signals or more complex methods do not generally obtain better results.

Finally, despite some studies that show that males and females have different physiological responses to stimuli[19], few works explore this.

2.6 Summary

In this section we took a look at some of the research that has been done in the area, and in specific, we analyzed how emotions are represented using the Circumplex Model of Affect, how EDA signals are frequently pre-processed, what kind of features are extracted, how they are selected and what are the most common estimators. We also highlighted the existence of some issues with the research, namely, the lack of attention to computing time and obtaining quick

responses, the use of few emotions and the lack of works that take differences in physiological responses between genders into account.

In the following chapter we describe the methods chosen for our algorithm, in order to overcome the issues related to computation time, number of emotions and gender specificity.

Citation	Signals	Pre-processing	Features and Feature Selection	Classes	Classifier/Estimator	Results
[34]	EDA	SCR/SCL separation, low pass filter (1.5Hz)	Temporal Features, Frequency Features, Morphological Features, Statistical Features; FS: ANOVA test	Calm/Distress	Decision Tree	89.18%
[4]	EDA,ECG,EOG, EEG,PPG	Linear detrend, windows of 20 seconds with 80% overlap	98 features in time and frequency domains; FS: Wrapper-based sequential forward selection to identify key subset of features	Relaxed/Excited	Medium KNN, Complex Tree, Medium Gaussian SVM, Bagged Tree	88.9% (100% for Excited state)
[29]	EDA, EEG	Resampling from 10Hz to 512Hz	Spectral lines in frequency domain; FS: t-test	Low Valence, High Valence, Low Arousal, High Arousal	KNN (k=3), SVM (with square kernel function)	72% (valence), 80% (arousal) with KNN
[13]	EDA, ECG	Digital Notch Filter (50Hz)	MP coefficients calculated after applying 3 different dictionaries to the signals (Coif5 at level 14, db4 at level 8, DCT); FS: LDA, PCA, Kernel PCA	5 classes (Happiness, Scary, Sadness, Peacefulness and Neutral). 3 distinct emotional categories (A - each class seperately; B - Low Valence/High Valence; C - Low Arousal/High Arousal)	Matching Pursuit	85.53% for GSR; 100% for ECG
[15]	EDA	Digital Notch Filter (50Hz), windows of 10 seconds with 50% overlap	Poincare plot, Recurrence quantification analysis, Entropy, Lyapunov exponents, Embedding dimension, Detrended flunctuation analysis, Kolmogorov, Triangle phase space mapping; FS: SFS, SFFS, RSFS	4 classes (Happiness, Scary, Sadness, Peacefulness)	LS-SVM, KNN, Quadratic, Fisher	87.5% (Fisher, any selection method)
[21]	EDA	Wavelet denoising, decomposition with db5 wavelet function	30 statistical features; Covariance analysis	5 classes (Happiness, Grief, Anger, Fear and Calm).	SVM	66.67%
[31]	EDA, ECG, EEG	Low-pass filter (3Hz)	32 EDA Statistical Features: SR, SC, SCSR, SCVSR; 77 ECG Features: spectral power, HRV, HR features; 105 EEG Features: PSD, Assymetry; FS: TPR, RCMSE, Shannon Entropy, XGBoost	Low Valence, High Valence, Low Arousal, High Arousal	SVM	80.1% (valence) 68.4% (arousal)
[12]	EDA, ECG	Welch's Poser Spectral Density, db6 at level 12, elliptical filters, Savitzky-Golay filter of order 12 frame size 75, FIR filter	Statistical Features, Power Spectral Density Features	Happy, Sad, Neutral	SVM, Naive Bayes, KNN	91% (EDA, KNN, PSD features)
[32]	EEG, ECG, HRV, RA, EDA	Not specified	36 EEG features, 33 ECG features, 28 RA features and 28 EDA features derived from analysis of time and frequency domains of each signal	Sadness, Happiness, Disgust, Fear, Neutral	SVM, Weighted Fusion	84.62%. Weighted Fusion Strategy improved result by about 10%
[35]	EDA, ECG, SKT, RSP, PD	Moving average filter, wavelet transform, windows of 10 seconds, signal decomposition by 6 levels of db5 wavelet transform	438 Physiological Features for each signal, wavelet transform, signal segmentation; FS: ANOVA test	Values of Valence and Arousal	LR, RR, SVR (3 different kernels) DT, KNN, MLP, NuSVR	MSE values of 0.07735 (valence) and 0.06499 (arousal) with NuSVR (nu = 0.5, C = 1.0)
[27]	ECG, EDA	Band-pass filters (0.05Hz - 19Hz)	Statistical Features	Low Valence, High Valence, Low Arousal, High Arousal	Deep Convolutional Neural Network, Naive Bayes, KNN, LDA, LSV, MLP, AdaBoost, Random Forest	EDA: 71% Arousal and 75% Valence with DCNN; ECG: 81% Arousal, 71% Valence with DCNN
[33]	EDA, EEG, ECG	Low-pass filter (60Hz)	60 Statistical Features for EDA; 50 Features for ECG based on HRV and Entropy; 378 Features for EEG based on Statistical, Hjorth, Entropy and Wavelets	Low/High Valence, Low/High Arousal	SVM, VAE	Arousal: 68.8%; Valence: 67%

Table 2.2: Table summarizing the methods used in works that use EDA.

Chapter 3

Algorithm for Estimation of the Emotional State

In Chapter 2 we discussed the methods commonly used by other researchers to estimate emotions. In this chapter we choose which methods to use. First we discuss the methodology used for the creation of the model. Afterwards, we define a number of requirements that real-time estimation imposes. Then we present the dataset with EDA signals used to tune the model and the metrics to evaluate it. Finally, we present the methods selected to be used in our algorithm.

3.1 Methodology

As mentioned in Chapter 2, emotional estimation algorithms follow a number of steps for their creation.

The first step is collecting the data. This can be done in one of two ways. The first is by conducting an experiment where people's physiological signals are collected while they receive some kind of stimulus, usually video, audio or image, and then are prompted to rate the emotion they felt. The second way is to use pre-existing datasets that already have this type of information. In our work, we will use pre-existing datasets.

The next steps, as described in Chapter 2 are: pre-processing, feature extraction and estimation. The methods were chosen by analyzing other works to find which methods are most common and have the best results. Then we choose the quickest of these, while performing tests to verify that they achieve good results. The methods chosen for each of these steps are detailed in the section Method Selection.

3.2 Real-time Estimation Requirements

As mentioned in Chapter 2, most research done has focused on developing techniques for identifying emotions from a small set of emotions and without paying attention to the time required to compute it. However, to achieve real-time estimation, there are several requirements that must be fulfilled:

- **Fast pre-processing**. For this, a windowing technique that extracts the signal in small segments (epochs) may be beneficial, as it can start pre-processing parts of the signal while it is still being captured. Other techniques that improve accuracy are potentially computationally expensive so it is required to ensure a good trade-off between speed and accuracy.

- **Fast feature extraction**. We can take advantage of the aforementioned windowing technique to start extracting features while the signal is being captured, as well as paralellization to calculate as many features as possible in parallel. As with pre-processing, some features may take too long to be computed, which should be avoided.

- **Fast estimation**. Estimation models such as Decision Trees and SVM may finish quickly, but algorithms such as Matching Pursuit may take quite a bit longer. Furthermore, the system as a whole must be very efficient to minimize processing time.

3.3 Dataset

The dataset chosen for the selection of features and the creation of the model was the 'A Dataset for Affect, Personality and Mood Research on Individuals and Groups' (AMIGOS)[2]. This dataset contains EDA, ECG and EEG signals from an experiment with 40 volunteers who watched 16 short videos. 27 of the participants were male and 13 were female. It provides ratings for Valence, Arousal, Dominance, Familiarity and Liking, as well as which basic emotion the person felt during a video out of seven possibilities: Neutral, Happiness, Sadness, Surprise, Fear, Anger, and Disgust. The ratings were collected through self-reporting. The values for Valence and Arousal were used in our work. In the dataset these values are in the range [1-9], but we recalculated them to fit the range [-0.5,0.5] to keep it consistent with other works.

The dataset is available in two forms: raw signals or pre-processed. In our work we used the raw version.

3.4 Metrics

To evaluate the quality of our algorithm we used the most common metrics to evaluate regression estimation (RMSE, MAE, PCC). In the following equations, y represents a series of N true values for the estimation, while $\hat{y}$ represents a series of N predictions from the estimator.

- Root Mean Square Error is commonly used to determine the accuracy of a regression model. Values closer to 0 mean the model has less deviation in its results;

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{N}}$$

- Mean Absolute Error tells the average of the error between the estimation and the expected (real) value. Values close to 0 mean that the average error is low;

$$MAE = \frac{\sum_{i=1}^{N} |\hat{y}_i - y_i|}{N}$$

- Pearson Correlation Coefficient is a measure of linear correlation. Values closer to 1 show that the model has strong linear correlation.

$$PCC = \frac{\sum_{i=1}^{N}(\hat{y}_i y_i) - \sum_{i=1}^{N} \hat{y}_i \sum_{i=1}^{N} y_i}{\sqrt{N \sum_{i=1}^{N} \hat{y}_i^2 - (\sum_{i=1}^{N} \hat{y}_i)^2} \sqrt{N \sum_{i=1}^{N} y_i^2 - (\sum_{i=1}^{N} y_i)^2}}$$

In addition, the estimated values will be converted into quadrants in the VA space, for which classification measures can be used to verify the accuracy of the algorithm at estimating the correct quadrant. The measures used for this are the following:

- Precision, which is the percentage of elements that were estimated for a class and truly belonged to that class;

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

- Recall, which is the percentage of elements that belong to a class and were estimated for the correct class;

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

- F1 Score, which is the harmonic mean of Precision and Recall;

$$F_1 = (\frac{Recall^{-1} + Precision^{-1}}{2})^{-1}$$

- Accuracy, which is the percentage of correctly classified elements.

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + FalsePositive + TrueNegative + FalseNegative}$$

Finally, time between stimulus and estimation response will be measured to evaluate if the algorithm can be used in real-time estimation.

3.5 Method Selection

In this section, we will look over the methods mentioned previously and choose the most adequate ones to achieve these requirements.

3.5.1 Pre-processing Methods

In Chapter 2 we described the most common pre-processing methods. To make the choice for our methods, we analyzed them, taking our purposes into account.

As mentioned before, one of our goals is to make the estimations in real-time. To achieve this we are unable to use the entire length of the signal, as that would only give an estimation at the end of the stimulus. In our case, we want several estimations over time. Typically, one every few seconds, to get a better idea of how a person feels at each point of the video. Researchers typically segment the signals in 10 or 20 seconds with 50% overlap, which we consider to be too long for localized estimations. On the other hand, a researcher using the AMIGOS dataset used a window of 2 seconds[28]. In the end, we chose a compromise length of 5 seconds with 50% overlap, as it provides a balance between the amount of data available and the speed at which estimations are done.

In addition to segmenting the signal, some noise removal or baseline removal can be done. SCR/SCL separation use a computationally expensive deconvolution operation[34], so we are not using it. We decided to use wavelet denoising, which smoothens the wavelet and attenuates odd frequencies in a similar way to a band-pass filter, without being computationally demanding. The most used wavelet family to denoise EDA signals is Daubechies. Although researchers use db4, db5 and db6, we found that db4 was the one that most accurately describes the EDA signal.

3.5.2 Features

In Chapter 2 we presented many kinds of features, of which we noted that the statistical time-based features are among the most used. These features obtain good results and also have the advantage that they are typically simple to calculate. A study on feature selection by Shukla, etc al.[28] showed that among different kinds of features, time-based statistical features are more likely to be chosen than other popular methods, losing out only to Mel- Frequency Cepstral Coefficient (MFCC) features. However, MFCC features are not as used and are more complex than statistical features. Thus, we chose to focus on statistical features as we have more works that use them and their calculation is faster.

In the end, the following features were chosen: maximum, minimum, range, standard deviation, skewness and kurtosis. The maximum and minimum are, respectively, the maximum and minimum values in the epoch. Range is the difference between maximum and minimum. Standard deviation, skewness and kurtosis were calculated with the following formulas, where s is global standard deviation, $\{x_1, x_2, ..., x_n\}$ are the sequence of values in the epoch, $\bar{x}$ is the average of the values in the epoch and N is the total number of values in the epoch:

$$StandardDeviation = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

$$s = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

$$Skewness = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^3/N}{s^3}$$

$$Kurtosis = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^4/N}{s^4}$$

To prevent some features having more or less importance than others, the features were normalized with the following formula, where x is the value of the feature and x' is the resulting normalized value.

$$x' = (\frac{x - mean(x)}{max(x) - min(x)})$$

In preliminary tests, these features obtained the results shown in Tables 3.1 and 3.2. Other features were also tested, like area under the curve and maximum absolute deviation, but they did not improve the results.

Valence	Both Genders
RMSE	0.065
MAE	0.012
PCC	0.973

Table 3.1: Preliminary results for Valence

Arousal	Both Genders
RMSE	0.051
MAE	0.010
PCC	0.973

Table 3.2: Preliminary results for Arousal

3.5.3 Estimators

One of the major problems presented in Chapter 2 was that most research focused on classifying emotions from a small set of emotions. To circumvent this problem, we use regression to estimate values for Valence and Arousal. As such, we use two estimators with the same features, one for Valence and another for Arousal.

We tested some of the most popular estimators. Namely, SVR, Random Forest and Decision Tree. As we can see in Tables 3.3 and 3.4, Decision Trees obtained the best results. The SVR results are surprising, given its popularity and general success in other works. However, many kernels and parameters were tested and these were the best results found. Random Forests were tested with number of trees up to 10000. Decision Trees were also tested with up to 10000 decision nodes.

Valence	SVR	Random Forest	Decision Tree
RMSE	0.281	0.254	0.065
MAE	0.240	0.218	0.012
PCC	0.087	0.498	0.973

Table 3.3: Preliminary results for the generic Valence Estimator.

Arousal	SVR	Random Forest	Decision Tree
RMSE	0.212	0.179	0.051
MAE	0.176	0.147	0.010
PCC	0.270	0.648	0.973

Table 3.4: Preliminary results for the generic Arousal Estimator.

3.6 Summary

In this chapter we stipulated some requirements for real-time estimation and chose the methods to use in our work, starting with the AMIGOS dataset to obtain the data, then the pre-processing of the signal, the feature extraction and the estimators. The chosen algorithm is as follows: first we segment the signal into five second windows with 50% overlap. Each window is then de-noised with a db4 wavelet. Then, the features maximum, minimum, range, standard deviation, skewness and kurtosis are extracted. Finally, the features are used to train and estimate from two Decision Trees, one for Valence and another for Arousal. The Decision Trees have a maximum of 10000 decision nodes.

While the methods chosen are not new and were used by many researchers, this combination has not been done before, due in part to our focus on real-time estimation.

We now need to build a framework with which our algorithm will be implemented, which we discuss in the next chapter.

Chapter 4

Emotional Recognition Framework

One of the problems mentioned in Chapter 2 was the lack of works relating to on real-time estimation. The requirements for real-time estimation mentioned in Chapter 2 presented a number of restrictions to the estimation algorithms we can use, but it also presents a challenge in how to structure the system to handle a fast estimation.

Our emotional recognition framework was conceived to tackle this challenge. The goal of this framework is to allow the user to build algorithms and experiments out of simple blocks that are all in execution simultaneously and can send data to the next blocks in the pipeline. With this system, the user can build algorithms using a sequence of blocks and use these to make experiments, and in particular, real-time experiments.

The following sections describe the arquitecture of the framework, the existing components and how to build our own emotion recognition system with it.

4.1 Framework Arquitecture

The resulting system created with the framework will be composed of several components that connect with each other to define an emotion estimation algorithm. These components are customizable, having the ability to input from or output to other components that may or may not be in use, depending on what the users need for their purposes. The connections between components are built with the following rule:

- Given component A already in the system and a component B, associate component A's output channel X to component B's input channel Y. Component B, if not already in the system, is added to it.

With this rule it is possible to create a variety of algorithms and experiments using these blocks. Some blocks may calculate more than one piece of information, but if the channels through which this data is sent are not connected to another component, there are mechanisms to skip their calculation, increasing performance further by only calculating what is needed.

A visual representation of the algorithm and its components that we will be building with this framework can be seen in Figure 4.1.

In the next section we will describe each component in more detail.

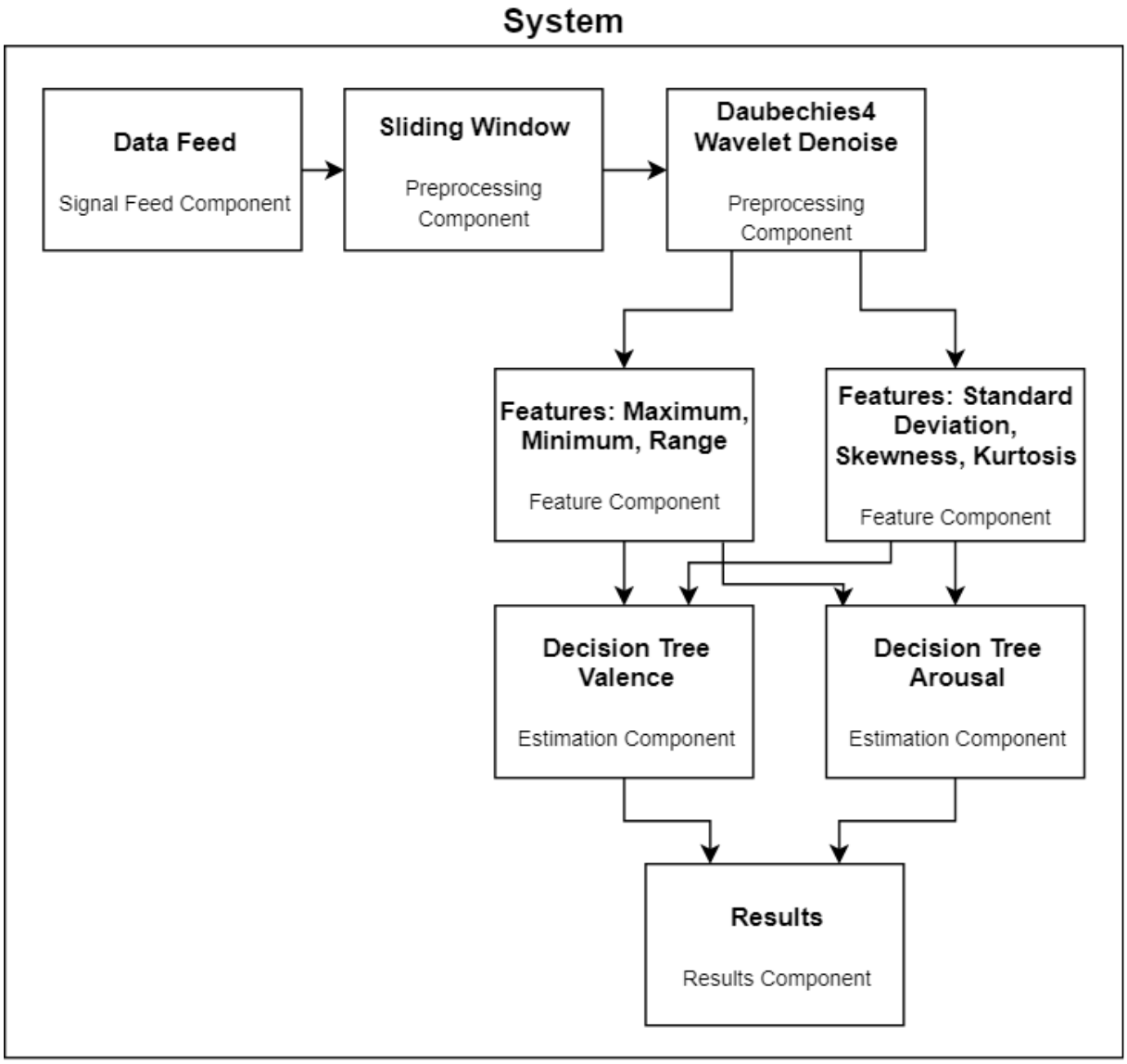

Figure 4.1: Diagram of the emotion recognition algorithm defined in Chapter 3 built in our framework. Estimators for gender-specific estimation omitted.

4.2 Components

From the research in Chapter 3 it was shown that emotion recognition algorithms tend to follow a similar flow - the signal is obtained, then preprocessed, then its features are extracted and finally the estimator is trained and used. The components of the framework, described in this section, follow this exact structure, making it simple for an user experienced in the area to understand where each component should be used.

- **Signal Feed** - This component receives signals from a data source, which can be a dataset or a device capable of extracting live physiological data, and passes it to the next components. It is also responsible for passing other information that may be necessary, for example gender, which may be necessary for experiments that have separated datasets depending on specific parameters not included in the signal itself.

- **Preprocessor** - This component is responsible for receiving the data sent by the Signal Feed and preparing it for the Feature Extractor. Cutting the signal into epochs and filters are examples of algorithms that fit in a Preprocessor.

- **Feature Extractor** - This component receives data from a Preprocessor or a Signal Feed (if the original signals are already preprocessed) and does all the necessary calculations to extract the features. The extracted features are passed to the Estimator using feature-specific channels, allowing extra information to be transmited.

- **Estimator** - In Training Mode, this component passes the feature vectors to its internal Dataset. When there is no more data to be processed, the Dataset is normalized and it can then be saved into a file to use later in Estimation Mode. In Estimation Mode, the component builds an estimator trained with the data in the loaded Dataset. Then, each feature vector received is normalized according to the data obtained when the Dataset was normalized in Training Mode and is then estimated using the estimator. This component is therefore responsible for maintaining its internal Dataset, training an estimator with machine learning techniques and performing the estimation to obtain the results. These are then passed to the Results component. It can also be used to perform validation, for example, using cross-validation.

- **Results** - This component contains the results of the estimation or validation performed by the Estimator. This is the 'output' component which gives the user results or statistics.

- **System** - This component controls the emotion recognition algorithm. It is responsible for forming the connections between components, starting and ending the execution of the components, and determining the Execution Mode.

4.3 Using the Framework

Using the framework is simple and requires only a few steps. The first step and most technical is the definition of the components. The second step is to use these components to define the algorithm, after which the algorithm can be executed. In the next sections we show some useful functions provided by the Component class and then show how to build the different kinds of components and how to build an emotion recognition system.

4.3.1 Component Class Functions

The Component parent class defines a few functions that are useful to control communication and execution. For communication, two simple functions are provided:

```
public void send(double value, int channel) throws ChannelException
public void send(double[] values, int channel) throws ChannelException
```

These functions, as their name indicates, send one value or a set of values to a specific channel. These functions are the mechanism through which data is sent to other components.

Several components can be defined to read from the same channel, as will be discussed in section 4.3.7.

```
public boolean isChannelOpen(int channel)
```

This function checks if a particular channel is in use by the algorithm. This is useful in several cases, for example, a Signal Feed component that is designed to send several physiological signals through their specific channels, but an algorithm that uses that component does not use all of its signals. Thus, by using this check, the unnecessary signals and calculations can be skipped.

```
public Queue getInputQueue(int channel) throws ChannelException
```

This is a vital function that gives a component access to its internal input queues. Each channel has an input queue where values are stored until the component retrieves them.

```
public void close()
public void closeIfNoInputs()
```

These functions close the component, terminating their execution. The first, *close()*, should be run as the final function in a component to mark it as closed. It can be seen as a forced shutdown in other situations. The second one marks the component as closed only if the components that transmit data to it are already closed and there are no more values in the input queues.

```
public boolean closed()
```

This function checks if the component is marked as closed and all of the components that transmit data to it are closed too. This control function is useful to determine if the component should stop executing.

4.3.2 Building Signal Feed Components

This component acts as the data entryway of the system. In order to build our own component, we must first take note of the constructor of the abstract class that defines a Signal Feed, which is as follows:

```
public SignalFeed(String name, int numOutChannels)
```

This constructor has two parameters: a name only has significance for error checking, such as a component that has not been properly set up, and an integer value that defines the number of exit channels the component has. This number defines how many channels the component can send data through.

To build our own Signal Feed component it needs to *extend* the SignalFeed class and define its own constructor which defines at least these parameters. The following is an example of a constructor for a component named AMIGOSFeedAllData.

```
public AMIGOSFeedAllData(String name, int[] windowsize, int[] slidesize)
{
    super(name, 7);
}
```

This component can be created by giving it a name and two arrays. These arrays are the size of the window and how many values are discarded after each window is complete, and these values are used to determine how many segments there are in each video.

Because the programmer knows how many channels the component will have, the value can be defined here. This component has seven channels, only three of which are used in this example (1 for the EDA signal, 3 for the expected arousal and 4 for the expected valence).

The next part of the definition of a component is the definition of the *run()* function.

```
@Override
public void run()
{
    //[...]
    for (int i = 0; i < NUM_USERS; i++)
    {
        //open user data file
        datastream = new BufferedReader(new FileReader(directory + "gsr" +
   (i+1) + ".csv"));

        for (int j = 0; j < NUM_VIDEOS; j++)
        {
            //get the number of samples in the video
            String line = datastream.readLine();
            String[] split = line.split(" ");
            double rows = Double.parseDouble(split[split.length - 1]);

            //get number of windows in the video
            size = (int)rows/slidesize[s]-(windowsize[s]/slidesize[s])-1;

            for (int z = 0; z < rows; z++)
            {
                //send value to next component
                line = datastream.readLine();
                split = line.split(" ");
               //send EDA
               send(Double.parseDouble(split[2]), 1);
            }

            for(int m = 0; m < size; m++)
            {
                //send self assessment values of arousal and valence,
   normalized from [1, 9] to [-0.5, 0.5]
                send(((selfassessment[i*NUM_VIDEOS+j][3] - 1) / 8) - 0.5,
   3);
                send(((selfassessment[i*NUM_VIDEOS+j][4] - 1) / 8) - 0.5,
   4);
            }
        }
    //[...]
    close();
}
```

The programmer can then define the code as they wish: either by extracting the data from a dataset or collecting it with hardware in real-time, using the APIs provided for the purpose. In this excerpt, the dataset was processed beforehand to facilitate its processing in the code.

To send data to the next components, the *send()* functions specified in section 4.3.1 are used.

At the end of the *run()* function the *close()* function should be run to mark it as closed.

4.3.3 Building Preprocessing Components

Building Preprocessing components is very similar to building Signal Feed components. The constructor of the abstract class is the following:

```
public ProcessingComponent(String name, int numInChannels, int
    numOutChannels, int numSamplesBaseline, int numSamplesStimulus)
```

This constructor has three more values. The first is numInChannels. Like numOutChannels is for the number of exit channels to send values to other components, numInChannels is the number of entry channels through which a component can receive values. The next two values are used to specify the number of samples that the signal has for baseline calculations and the number of samples of stimulus. Depending on the algorithm, these two may not be used.

The following is an example of a constructor for a component named Daubechies:

```
public Daubechies(String name, int numSamplesBaseline, int
    numSamplesStimulus, int wavelet)
{
    super(name, 1, 1, numSamplesBaseline, numSamplesStimulus);
    this.wavelet = wavelet;
}
```

Like with Signal Feed components, the number of entry and exit channels is known to the programmer. In this case an extra parameter is added, making the system engineer able to create any of the different kinds of Daubechies wavelets without having to build a different component.

Also like Signal Feed, only the *run()* function is left to define. However, in addition to using *close()* at the end of the function, the *run()* function should have a loop like the following, which allows the component to execute until it is either forcibly closed or there is nothing left for it to do.

```
@Override
public void run()
{
    while(!closed())
    {
        // code here

        closeIfNoInputs();
    }
    close();
}
```

4.3.4 Building Feature Components

Building Feature components is similar to Preprocessing components. The constructor of the abstract class has one difference:

```
public FeatureComponent(String name, int numInChannels, int nfeatures, int
    numSamplesBaseline, int numSamplesStimulus)
```

In the Features component, numOutChannels changes to nfeatures. The name change does not affect how the component is used and merely reflects what the component itself is used. As such, building the Features component is identical to the Preprocessing component, with the exception that the Feature component can connect to the Estimation component's feature channels.

Like numInChannels and numOutChannels in the previous two components, nfeatures is known to the programmer, so it can be defined by default, as presented in the next example of a component called MaxMinRanFeature, which calculates the maximum, minimum and range features.

```
public MaxMinRanFeature(String name, int numSamplesBaseline, int
    numSamplesStimulus)
{
    super(name, 1, 4, numSamplesBaseline, numSamplesStimulus);
}
```

These components should have a structure for the *run()* function equal to the Preprocessing components.

4.3.5 Building Estimation Components

Building the Estimation components has a few more steps. The constructor of the abstract class is similar to the previous components:

```
public EstimationComponent(String name, int numInChannels, int
    numOutChannels, String filename)
```

Likewise, the construction of an Estimation component called RegressionTreeEstimator can be done in this way:

```
public RegressionTreeEstimator(String name, int numInChannels, int
    numOutChannels, String filename)
{
    super(name, numInChannels, numOutChannels, filename);
}
```

The Estimation Component requires the definition of more functions other than *run()*, namely, *train()*, *estimate()* and *validate()*.

Another useful, already defined function is *fillDataset()*, which is used to fill the estimator's internal dataset prior to the training procedure. While the training could be done at the same time, it would offer no advantage as the training process is done separetely from the estimation process, thus execution in real-time is not needed. In this way we can build the entire dataset which we can dump to a file (*dump()*) and restore it later to use it *restore()*. The file used for the *dump()* and *restore()* functions is the file specified in the *filename* parameter.

An example of *run()* function is as follows:

```
@Override
public void run()
{
    while(!closed())
```

```
    {
        super.run();
        if (allFeaturesAvailable())
        {
            switch(getEstimationState())
            {
                case FILLDATASET:
                    if(!hasFillFinished() && !getClassQueue().isEmpty())
                    {
                        fill_dataset();
                    }
                    break;
                case ESTIMATE:
                    estimate();
                default:
                    break;
            }
        }
    }
}
```

super.run() executes code already defined to automatically gather the features from the feature channels. Then the *allFeaturesAvailable()* check determines if all the features for the current data point are ready. Then, depending on the Estimation System's state, which will be discussed in Section 4.3.7, it either fills the dataset or estimates. If it is filling the dataset, it makes sure that the truth value is available as well before doing so.

The code above does not use *train()* or *validate()*. While it could use it, both of these functions are run few times in a controlled way, unlike *fillDataset()* which puts one data point into the dataset and *estimate()* which estimates one point of data every time they are executed. As such it is more useful to execute them elsewhere.

When programming the *estimate()* and *validate()* functions, it is possible to use them to compile the results. However, doing this prevents the results from being combined with other estimators, making some estimation methods such as ensemble harder. To work around this, the Results component was introduced, which will be discussed in the next section.

4.3.6 Results Component

The Results component acts as a result processor and repository. The constructor of the abstract class is as follows:

```
public Results(String name, int numInChannels)
```

Other than the *run()* function which should be used to gather the results from the various channels, this component requires the defiinition of two other functions: *printAllResults()* and *getLatestResult()*, which serve the purposes implied by their names.

As an example, a component called VAResults and its constructor and mandatory functions:

```
public VAResults(String name)
{
    super(name, 4);
    truearousal = new ArrayList<>();
```

```
    truevalence = new ArrayList<>();
    predarousal = new ArrayList<>();
    predvalence = new ArrayList<>();
    latestV = Double.NaN;
    latestA = Double.NaN;
}
@Override
public void printAllResults()
{
    for(int i = 0; i < predarousal.size(); i++)
    {
        System.out.println("VA: ("+predvalence.get(i)+","+ predarousal.get(
    i)+")");
    }
}
@Override
public String getLatestResult()
{
    return "VA: ("+latestV+","+ latestA+")";
}
@Override
public void run()
{
    while(!closed())
    {
        try
        {
            Double r = null;

            // check each input queue for predicted inputs and update
    latest prediction values if needed
            if ((r = (Double) getInputQueue(0).poll()) != null)
            {
                predvalence.add(r);
                if (predvalence.size() <= predarousal.size())
                {
                    latestA = predarousal.get(predvalence.size() - 1);
                    latestV = predvalence.get(predvalence.size() - 1);
                }
            }
            if ((r = (Double) getInputQueue(1).poll()) != null)
            {
                predarousal.add(r);
                if (predarousal.size() <= predvalence.size())
                {
                    latestA = predarousal.get(predarousal.size() - 1);
                    latestV = predvalence.get(predarousal.size() - 1);
                }
            }

            //check each input qeue for expected values
            if ((r = (Double) getInputQueue(2).poll()) != null)
                truevalence.add(r);
            if ((r = (Double) getInputQueue(3).poll()) != null)
                truearousal.add(r);

        }
        catch (ChannelException e)
```

```
            {
                System.out.println(e.getMessage());
            }
        }
        close();
}
```

The expected values in this component serve to gather statistics from estimation or validation from the Estimator. Different functions can be defined to gather statistics and metrics, depending on what is needed to get the results from the algorithm.

4.3.7 Building the System

With all the components defined, the system can be built. We start by defining the system itself. In this case, we create the emotion recognition system and set it to fill the dataset. The following code define the system shown in Figure 4.1.

```
EmotionEstimationSystem e = new EmotionEstimationSystem();
e.setState(EstimationState.FILLDATASET);
```

Then we define each component and connect them with each other.

```
SignalFeed feed = new AMIGOSFeedAllData("AMIGOS Feed");
e.addSignalFeed(feed);
SlidingWindow sdgsr = new SlidingWindow("SlidingWindowEDA", WINDOWSIZE_EDA,
    SLIDESIZE_EDA);
e.addProcessingComponent(feed, sdgsr, 1, 0);
```

The *addSignalFeed()* function adds the *feed* to the system and *addProcessingComponent()* adds the *sdgsr* component to the system and also connects the exit channel 1 of *feed* to the entry channel 0 of *sdgsr*. The same pattern follows for the rest of the components. *WINDOWSIZE_EDA* and *SLIDESIZE_EDA* are constant values defined beforehand that tell the Sliding Window how many values to use for the windows and how many values are discarded each slide.

```
final int WINDOWSIZE_EDA = 5*128;
final int SLIDESIZE_EDA = WINDOWSIZE_EDA/2;
```

These values are equivalent to windows of five seconds at 128Hz with 50% of the values being discarded each slide.

```
Daubechies d4 = new Daubechies("DB4", 0, WINDOWSIZE_EDA,4);
e.addProcessingComponent(sdgsr, d4, 0, 0);
MaxMinRanFeature mmrf = new MaxMinRanFeature("MaxMinRan", 0, WINDOWSIZE_EDA
    );
SDSkewKortMADFeature sdskk = new SDSkewKortMADFeature("SDSkewKurtMAD", 0,
    WINDOWSIZE_EDA);
e.addFeatureComponent(d4, mmrf, 0, 0);
e.addFeatureComponent(d4, sdskk, 0, 0);
```

In thes last two lines we see that both Feature components are connected to *d4*'s exit channel 0. This allows both components to receive the same data for processing.

Then we define the estimators and add features to them. In this case, the first three features of each Feature Component are added to the two estimators.

```
RegressionTreeEstimator arousalGeneral = new RegressionTreeEstimator("
    AMIGOSREGTREEAROUSAL", 0, 2, "AMIGOSREGTREEAROUSAL");
RegressionTreeEstimator valenceGeneral = new RegressionTreeEstimator("
    AMIGOSREGTREEVALENCE", 0, 2, "AMIGOSREGTREEVALENCE");
e.addFeaturesToEstimator(mmrf, arousal, new int[]{0, 1, 2});
e.addFeaturesToEstimator(mmrf, valence, new int[]{0, 1, 2});
e.addFeaturesToEstimator(sdskk, arousal, new int[]{0, 1, 2});
e.addFeaturesToEstimator(sdskk, valence, new int[]{0, 1, 2});
```

In addition, we connect *feed* to the estimators through a special channel, which has the responsibility to provide the real values for the estimation in order to train the model or, in the case of estimation, compare results to the expected values.

```
e.setEstimatorClassInput(feed,arousalGeneral, 3);
e.setEstimatorClassInput(feed,valenceGeneral, 4);
```

Finally, we create the Results component and tell the estimators to send the results. The top two *addResults()* send the predicted values and the bottom two send the expected values for comparison.

```
VAResults va = new VAResults("ResultsVA");
e.addResults(arousalGeneral, va, 0, 1);
e.addResults(valenceGeneral, va, 0, 0);
e.addResults(arousalGeneral, va, 1, 3);
e.addResults(valenceGeneral, va, 1, 2);
```

With the model now built, we can run experiments. To fill the dataset, the following code can be run:

```
e.startAllComponents();
while(!e.finishedFillingDataset())
{
    Thread.sleep(1000);
}
e.normalizeDatasets();
e.dump();
e.testModels();
e.stopAllComponents();
```

First the components are started, and then we wait for the dataset filling process to end, which is defined as the moment all estimators are marked as closed, after which we normalize the datasets and dump it into a file for use later. Then we can test the models, which runs the *validate()* function for all Estimation components. Finally we stop all the components and can now get results from the Results component if any functions to collect statistics have been specified.

To run an estimation experiment, the process is similar:

```
arousal.restore();
valence.restore();
arousal.train();
valence.train();
e.startAllComponents();
while(!e.over())
{
```

```
    Thread.sleep(1000);
    System.out.println(va.getLatestResult());
}
e.stopAllComponents();
```

First, since we dumped the datasets to files before, we can just load them again with *restore()*. Then we use *train()* to build the models and start the experiment. In this case, during the estimation process we print out the last estimation for Valence and Arousal every second. The process ends when there is no more data flowing in to the estimators due to all the components before them being closed, which we determine with the *over()* check.

4.4 Tecnologies

The framework was built using the Java language and the library Statistical Machine Intelligence & Learning Engine (SMILE)[1]. This Java library offers a wide variety of algorithms for machine learning, model validation, statistics, and even some common preprocessing tecniques like wavelet transforms that many researchers use, making it a better choice for this project than alternatives like Weka[2].

4.5 Summary

In this chapter we described a framework for building emotional recognition algorithms. The framework uses small blocks that communicate with each other and execute in paralell to increase algorithm efficiency. The blocks represent the several steps of the general approach to emotional recognition. With the SignalFeed, the signals are taken from a data provider and are input into the system. Then the Preprocessor is responsible for preparing the signals for feature extraction, which is done by the Feature Extractor. Finally, the features are used to train or estimate from an Estimator, from which results are compiled in the Results component. We also presented an example of use of the framework.

With the framework built, we can implement our algorithm which was defined in Chapter 3. In the next chapter we present the results of our experiments.

[1] https://haifengl.github.io/smile/

[2] https://www.cs.waikato.ac.nz/ml/weka/

Chapter 5

Evaluation

We implemented the algorithm described in Chapter 3 using our the framework detailed in Chapter 4, and executed a set of experiments. In the next sections, we describe the evaluation, starting with the methodology used, then we describe the datasets and the metrics used. Afterwards we present the results obtained and discuss them.

5.1 Methodology

To evaluate our algorithm for estimation of the emotional state of people, we performed several experiments. The first experiment was done with the AMIGOS dataset and it trained the two estimators, one for Valence and another for Arousal, with data from people of both genders. Then, two sets of estimators were trained, each set with data only from people from one of the genders. 10-fold cross validation was used to extract the metrics in order to evaluate the models. In addition, we converted the estimated values of Valence and Arousal to evaluate the accuracy of the binary classification of High/Low and to classify the quadrants of the Circumplex Model of Affect, which can be seen in Figure 5.1. This is done to compare our accuracy it with the accuracy of other works that use classification. Afterwards, identical experiments were done with the DEAP dataset.

A final experiment was done where values from the signal is input every 1/128th of a second, simulating a real-time experiment, in order to determine how long the algorithm takes to estimate in a real-time situation.

5.2 Datasets and Metrics

In addition to the dataset described in Section 3.3, AMIGOS, we also use 'A Database for Emotion Analysis using Physiological Signals' (DEAP)[18] for evaluation. This dataset provides EEG, EDA, Respiration Rate, Plethysmograph and Temperature signals from an experiment where 32 volunteers, 17 of which were male and 15 female, watched 40 music videos. It provides self-assessed ratings for Valence, Arousal and Dominance. Like AMIGOS, the values are in the range of [1-9], so they were recalculated into the range of [-0.5,0.5]. The dataset comes in two forms: raw or pre-processed. In our case we used the pre-processed version of

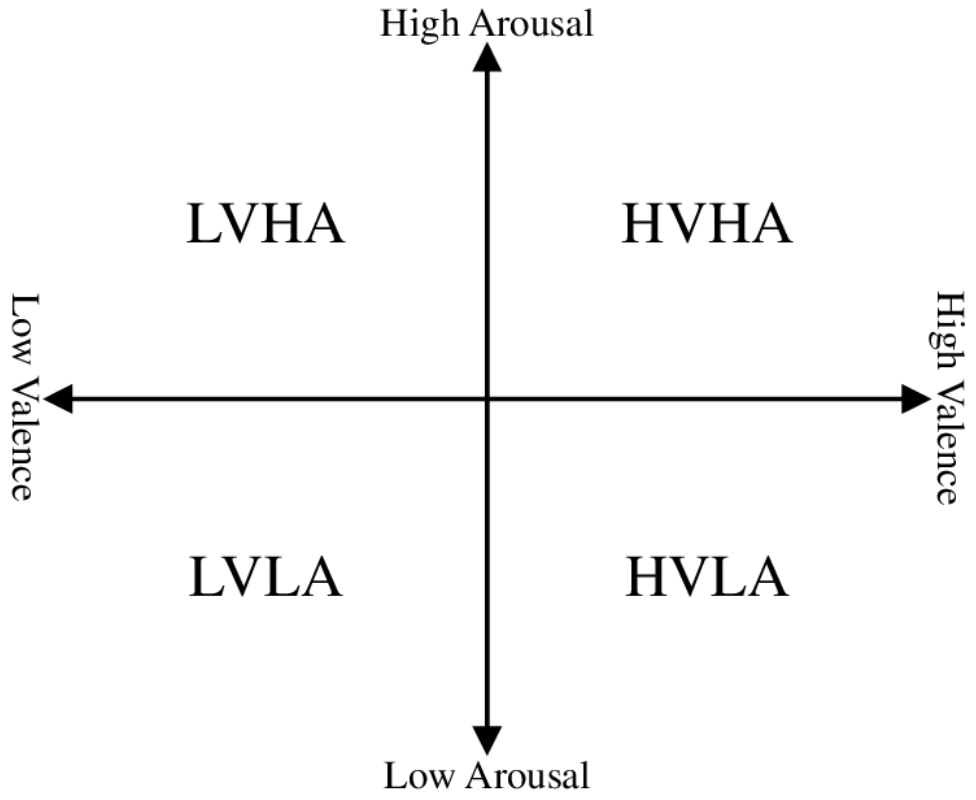

Figure 5.1: The quadrants in the Circumplex Model of Affect.

this dataset. Due to the preprocessing, the EDA signal was heavily changed, which allows us to test the robustness of our algorithm for different EDA signal processing methods.

The metrics used to evaluate the algorithm are the same as the ones described in Section 3.4.

5.3 Results

The results are divided into three sections. In the first section we present the results for the AMIGOS dataset, while on the second we present the results for the DEAP dataset. Inside each dataset section, the results for estimation and classification are separated. In the third section we present the results on performance, which are independent of the dataset.

5.3.1 Results for the AMIGOS Dataset

A total of six estimators in three sets of two were trained. Each set has one Decision Tree for Valence and another for Arousal. The first set was trained with data from both genders and the second and third sets were trained with data from only male and only female, respectively.

5.3.1.1 Estimation Results

The results of the 10-fold cross-validation are as presented in Tables 5.1 and 5.2.

These values show that the algorithm works very well for the AMIGOS dataset. Values of MAE show that the average error is very low and the RMSE shows that there is little variance.

Valence	Both Genders	Male	Female
RMSE	0.061	0.056	0.067
MAE	0.011	0.009	0.012
PCC	0.976	0.981	0.973

Table 5.1: Estimation results for Valence on the AMIGOS dataset.

Arousal	Both Genders	Male	Female
RMSE	0.049	0.045	0.070
MAE	0.009	0.007	0.018
PCC	0.974	0.979	0.951

Table 5.2: Estimation results for Arousal on the AMIGOS dataset.

The estimation for Male-only obtained slightly better results than with the estimation with both genders, while Female-only performed slightly worse. Figures 5.2 and 5.3 show a heatmap[1] of the occurrences of each prediction compared to the expected value for the estimation with data from both genders, for Valence and Arousal respectively. Because the predictions are floating point values, the scale [-0.5,0.5] was converted to a discrete scale with 100 positions, so results with deviation less than 0.01 are considered equal in the heatmaps. The color scale is from cyan (0 predictions in that area) to red (maximum amount of predictions). However, the scale for the colors is logarithmic to give better visibility for the lower frequency prediction zones.

As we can see from the heatmaps, most of the predictions are on the $y = x$ line, meaning that predictions tend to be equal or very close to the expected value, as also indicated by the PCC values.

5.3.1.2 Classification Results

After obtaining the values for Valence and Arousal from the estimation, we determined if the values are negative or positive to get accuracy rates for High/Low estimation. In addition, we use the same technique to classify the quadrants in the Circumplex Model of Affect.

Accuracy	Both Genders	Male	Female
Valence	0.982	0.984	0.980
Arousal	0.980	0.988	0.964

Table 5.3: Accuracy for High/Low classification for the AMIGOS dataset.

In Table 5.3 we can see that Male achieves better results for the binary classification of High/Low, while Female achieves the worst, particularly in Arousal. This behaviour is consistent with the results in Tables 5.1 and 5.2.

After obtaining the quadrants we can see the effects of the previous results in Table 5.4. We can see that our algorithm is able to correctly estimate the quadrant with an accuracy of 96.3%, 97.3% and 94.5% for Both Genders, Male and Female respectively.

[1] Generated with JHeatChart - http://www.javaheatmap.com/

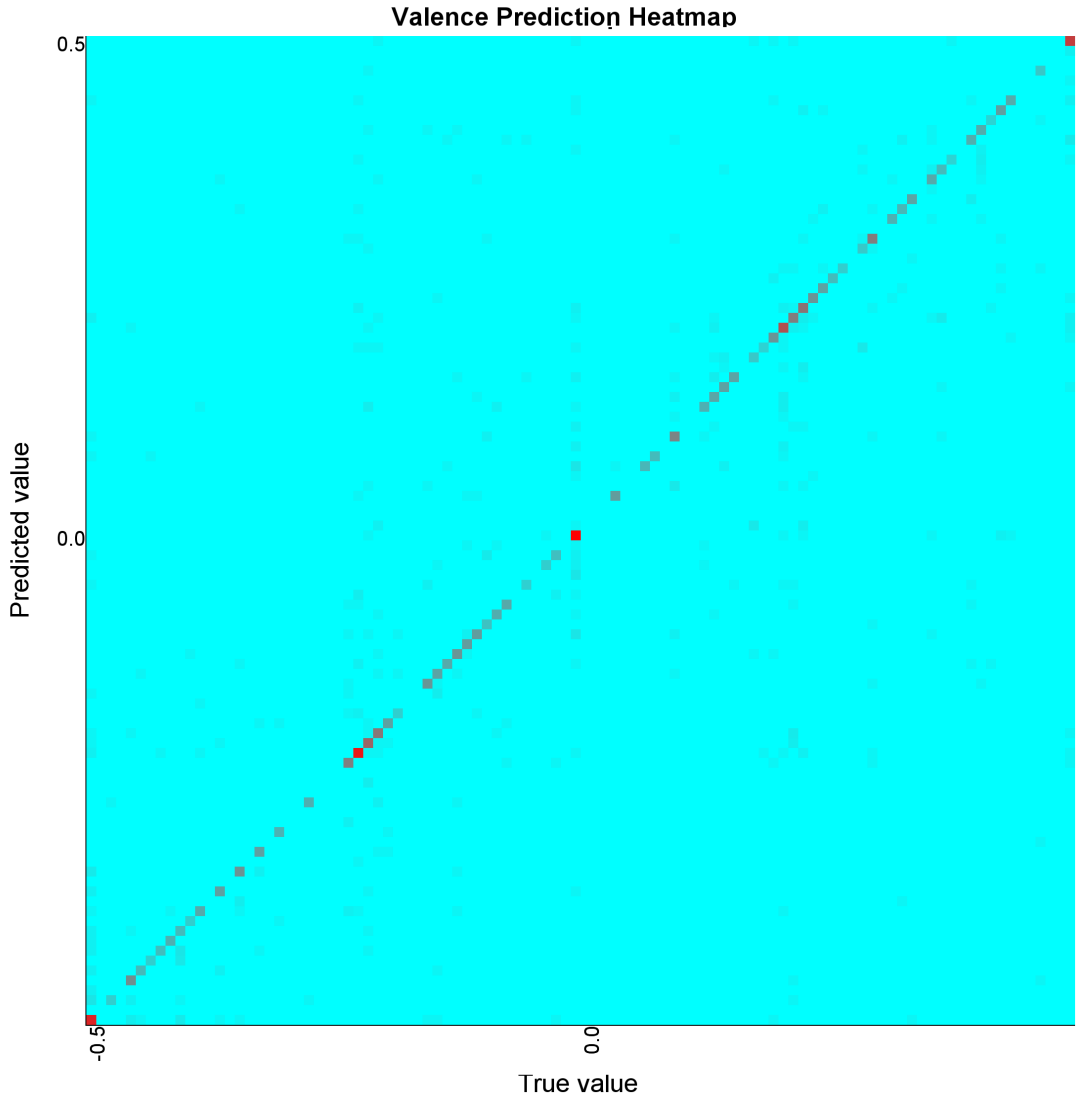

Figure 5.2: A heatmap of the true value against the predicted value for Valence for the AMIGOS dataset.

	Both Genders	Male	Female
Precision	0.964	0.973	0.945
Recall	0.960	0.971	0.942
F1	0.962	0.972	0.944
Accuracy	0.963	0.973	0.945

Table 5.4: Classification Results for the AMIGOS dataset.

Tables 5.5, 5.6 and 5.7 show the Confusion Matrices for the three gender experiments. These Confusion Matrices are consistent with the results above, and also show that when the algorithm predicts an incorrect quadrant for Arousal, it is very unlikely to also predict an incorrect quadrant for Valence, and vice-versa.

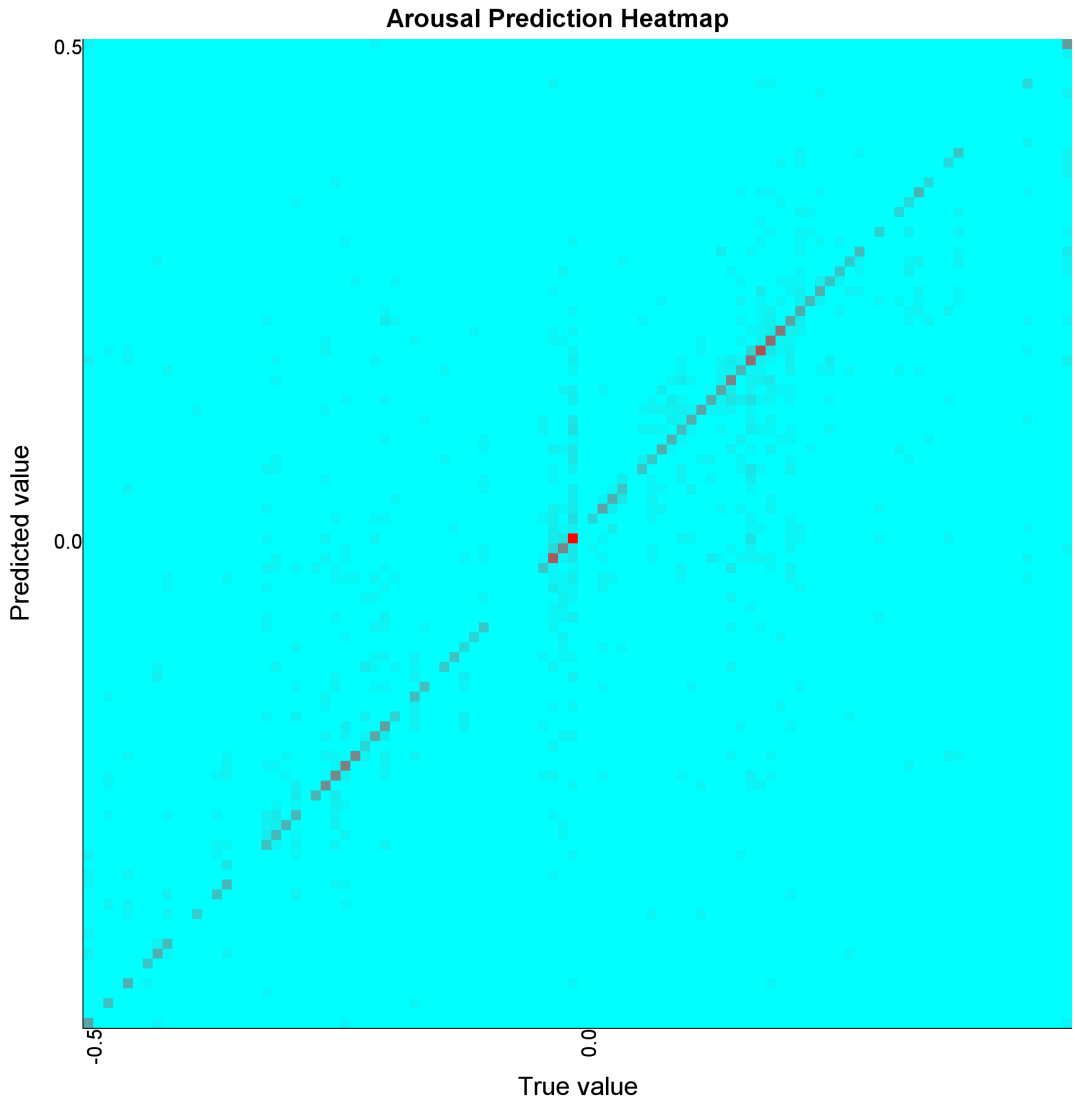

Figure 5.3: A heatmap of the true value against the predicted value for Arousal for the AMIGOS dataset.

Both Genders	LVHA	HVHA	LVLA	HVLA
LVHA	5870	82	121	2
HVHA	152	6643	3	128
LVLA	69	1	3532	47
HVLA	1	71	90	3975

Table 5.5: Confusion Matrix for Both Genders in the AMIGOS dataset.

Male	LVHA	HVHA	LVLA	HVLA
LVHA	3963	49	49	0
HVHA	100	4620	0	52
LVLA	32	2	2260	27
HVLA	1	29	40	2634

Table 5.6: Confusion Matrix for Male in the AMIGOS dataset.

Female	LVHA	HVHA	LVLA	HVLA
LVHA	1919	31	68	5
HVHA	50	2016	3	76
LVLA	43	0	1257	19
HVLA	0	55	32	1355

Table 5.7: Confusion Matrix for Female in the AMIGOS dataset.

5.3.2 Results in the DEAP Dataset

Like the AMIGOS dataset, the three sets of two Decision Trees for each gender group were trained and then used for 10-fold cross validation.

5.3.2.1 Estimation Results

The results shown in Tables 5.8 and 5.9 are not as good as in AMIGOS. In this case, both gender-specific estimations obtained better results compared to Both Genders in all the metrics for both Valence and Arousal.

Valence	Both Genders	Male	Female
RMSE	0.127	0.112	0.122
MAE	0.055	0.047	0.049
PCC	0.882	0.900	0.901

Table 5.8: Estimation results for Valence on the DEAP dataset.

Arousal	Both Genders	Male	Female
RMSE	0.127	0.107	0.116
MAE	0.056	0.044	0.046
PCC	0.868	0.909	0.901

Table 5.9: Estimation results for Arousal on the DEAP dataset.

The difference between the AMIGOS dataset and the DEAP dataset experiments are highlighted by the heatmaps in Figures 5.4 and 5.5. These heatmaps show more errors in estimation than their AMIGOS counterparts in Figures 5.2 and 5.3

5.3.2.2 Classification Results

Like with AMIGOS, the estimated values of Valence and Arousal were used to classify High-/Low Valence and Arousal and the quadrant in the Circumplex Model of Affect.

Accuracy	Both Genders	Male	Female
Valence	0.914	0.924	0.924
Arousal	0.902	0.922	0.920

Table 5.10: Accuracy for High/Low classification for the AMIGOS dataset.

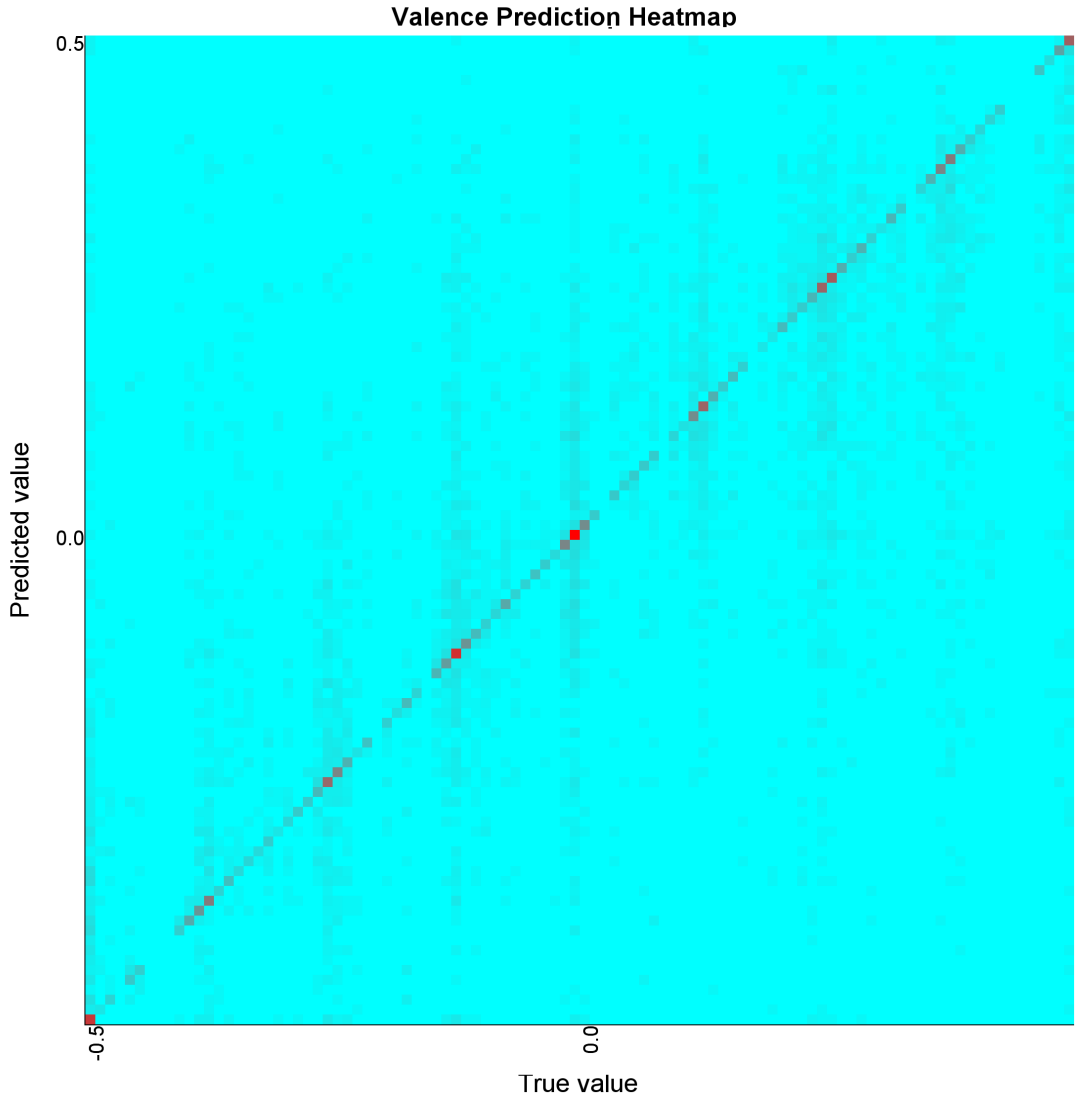

Figure 5.4: A heatmap of the true value against the predicted value for Valence for the DEAP dataset.

	Both Genders	Male	Female
Precision	0.822	0.847	0.848
Recall	0.820	0.846	0.847
F1	0.820	0.846	0.847
Accuracy	0.824	0.852	0.849

Table 5.11: Classification Results for the DEAP dataset.

In Table 5.10 we can see that the results overall worse than in the AMIGOS dataset, as expected from the estimation results.

Likewise, the increase of error rates in Valence and Arousal makes the results of the quadrant classification significantly lower, as seen in Table 5.11.

Tables 5.12, 5.13 and 5.14 show the Confusion Matrices for the three gender experiments. These matrices, in comparison to the ones in AMIGOS, show more tendency to get both pre-

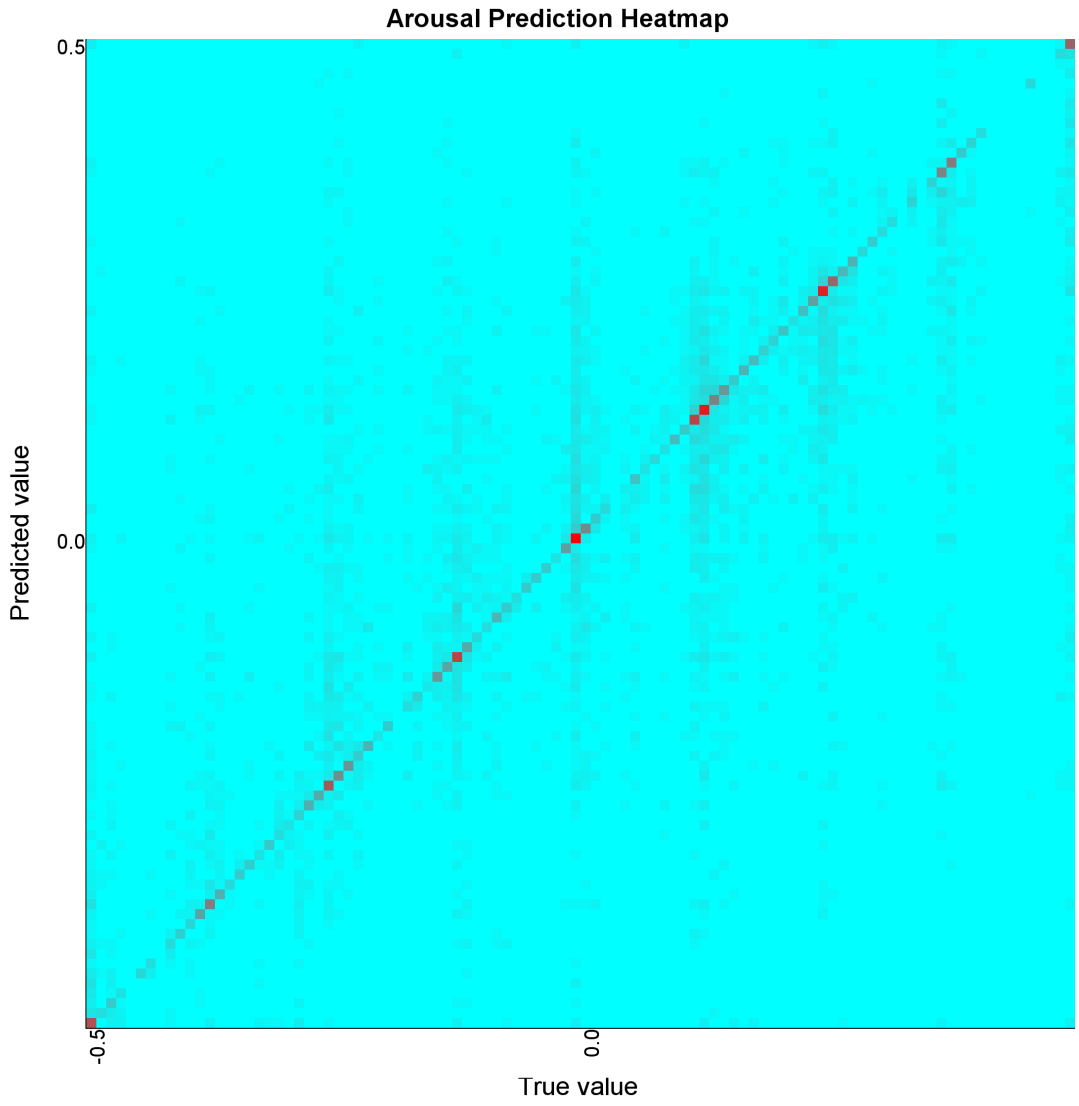

Figure 5.5: A heatmap of the true value against the predicted value for Arousal for the DEAP dataset.

dictions wrong, though it is still more likely to predict one correctly than none.

Both Genders	LVHA	HVHA	LVLA	HVLA
LVHA	6444	706	619	72
HVHA	708	8602	69	876
LVLA	554	59	4414	476
HVLA	63	713	477	5868

Table 5.12: Confusion Matrix for Both Genders in the DEAP dataset.

5.3.3 Performance

The final aspect for evaluation is how fast the estimation is done. To do this, the time between the last input of a signal epoch and the estimation reaching the Results component was mea-

Male	LVHA	HVHA	LVLA	HVLA
LVHA	3229	367	245	28
HVHA	316	5049	30	384
LVLA	207	14	2155	235
HVLA	22	344	229	3466

Table 5.13: Confusion Matrix for Male in the DEAP dataset.

Female	LVHA	HVHA	LVLA	HVLA
LVHA	3489	295	254	23
HVHA	313	3757	18	299
LVLA	249	21	2350	183
HVLA	18	270	226	2635

Table 5.14: Confusion Matrix for Female in the DEAP dataset.

sured. The maximum time was 16ms, with most values below 5ms. Further analysis showed that the maximum time was spent on the first estimation. This may be due to the way Java handles threads.

Because the time spent in each estimation is very low, the algorithm can be used in real-time.

5.4 Discussion

The results obtained with the AMIGOS dataset are very good, with a very low RMSE and MAE meaning that any errors that exist are usually very small. There was little difference between the gender-specific and the general estimation in the AMIGOS dataset, but in DEAP the difference is bigger, with both Male and Female getting better results than Both Genders. The reduced accuracy in the Female model of the AMIGOS experiment may be because of the low amount of data to train the model. In the AMIGOS dataset, there are 27 male participants and 13 female participants, while in the DEAP dataset Male having 17 and Female has 15. It is possible that with more data for female participants in the AMIGOS dataset a more robust model would be obtained that would also improve over the result for Both Genders.

The results in DEAP are not as promising as in AMIGOS, but there are a number of factors involved. The algorithm was tuned and tested first with the AMIGOS dataset and we used the pre-processed version of the DEAP dataset, which heavily altered the signals. With that in mind, the results for DEAP can be seen as proof that the algorithm works fairly well at handling different kinds of data.

Our work, unlike many authors, estimated values of Arousal and Valence. However, other authors did High/Low classification. We can see in Table 5.15 a comparison between our results and theirs.

Zhang, et al.[35], like us, estimated values of Valence and Arousal. As we can see in Table 5.16, we obtained much better results.

In addition to good results in Arousal and Valence, the combined results are also very good, achieving 96% accuracy in quadrant estimation, which is better then the results other researchers

Accuracy	Valence	Arousal
Tarnowski, et al.[29]	0.720	0.800
Wei, et al.[32]	0.710	0.750
Yang, et al.[33]	0.690	0.670
Our Work (DEAP)	**0.914**	**0.902**
Our Work (AMIGOS)	**0.982**	**0.980**

Table 5.15: Comparison between our results for binary High/Low classifications and other researchers.

RMSE	Valence	Arousal
Zhang, et al.[35]	0.278	0.255
Our Work (DEAP)	**0.127**	**0.127**
Our Work (AMIGOS)	**0.061**	**0.049**

Table 5.16: Comparison between estimation results. MSE values presented in Zhang, et al.'s work[35] were recalculated into RMSE.

obtained, as seen in Table 2.2. This may imply that estimating values of Valence and Arousal and using these obtained values to classify may be better than doing classification for arbitrary sets of emotions.

Finally, the estimation time is below 20ms, which is very quick. Therefore this algorithm, with help from the framework, enables real-time estimation with good results.

5.5 Summary

In this chapter we presented and discussed the results of our evaluation. We concluded that our results are very good for the AMIGOS dataset, both for estimation, which has very low error rates, and classification, which due to the low error rates obtained 96% accuracy on the general experiment. For the DEAP dataset the results were a bit worse (82% accuracy), but the accuracy was still good in comparison with many other works.

The results for gender-specific estimation were not significantly different to the general estimation, but in the DEAP dataset both Male and Female achieved better results on all metrics. In the AMIGOS dataset, Male achieved better results and Female achieved worse results. Since the worse results for Female in AMIGOS may be due to the relatively low amount of data in comparison with the others, it may be beneficial to create separate gender models when there is a good amount of data for both genders.

The algorithm runs efficiently, with the longest time between end of signal and estimation being 16ms. Therefore, the algorithm can run in real-time with good accuracy rates.

Chapter 6

Conclusions and Future Work

In this chapter we take some final conclusion from our work. We start by summarizing the document, then we discuss our contributions and the limitations of our solution. Finally, we end discussing possible future work.

6.1 Summary of the Dissertation

In Chapter 1 we presented a brief overview of the importance of emotion recognition. We noted that some key issues exist with current research and presented four main goals to achieve during the course of the work, namely, the research of current methods for emotion recognition with physiological signals, the design and implementation of a framework for emotion recognition, the implementation of an algorithm in the framework and the construction of experiments to validate the algorithm and determine if separating genders improves the results.

In Chapter 2 we delved into the research, finding several methods other researchers use to determine the emotional state, in particular, how they represent emotions and how they pre-process EDA signals, extract its features, and perform classification or estimation. In doing so, we got a further understanding of the issues to solve.

In Chapter 3 we analyzed the results of our research and determined the best course of action to build an algorithm for emotion recognition. We did so by first establishing some requirements, choosing a dataset to tune our model and choosing metrics to evaluate it. Then we decided on the methods to use for our algorithm for it to be accurate and efficient.

In Chapter 4 we described our framework, an important tool that ensures our algorithm executes as efficiently as possible. We preented its arquitecture, the components that compose it and how it is used.

Finally, in Chapter 5 we presented the results of our evaluation. For our experiments we used two datasets, which obtained different results. with the AMIGOS dataset, we achieved very good results, with low values in deviation and error and high values in correlation, resulting in the correct prediction of quadrants in 96% of the cases on the general experiment. The results with the DEAP dataset were also good, with a accuracies above 82%. In the case of DEAP, gender-specific estimators performed better by around 2.5%.

6.2 Contributions and Limitations

This work has two main contributions. The first is a new algorithmic process for emotion recognition. While the process itself is new, it uses a combination of already existing and well known methods, chosen to execute fast in order to perform the recognition in real-time while getting good results at the same time. The results of our experiments show that our estimations have low deviation from the expected values, achieving high accuracies. However, using a different dataset the results were slightly worse, but still very good. This, however, can be considered a limitation of our work. As different datasets contain different signal data due to differences in physiological signal monitors or signal processing, our algorithm is not guaranteed to always provide the good results presented in this work.

The second contribution is a framework that allows building complex emotion recognition algorithms by combining smaller blocks of code which communicate in a simple and fast way and run in paralell, making it possible to perform estimation at the same time as other epochs are being processed by other parts of the code, improving efficiency and allowing real-time estimation. However, the framework has two limitations. The first is the memory. If there is too much data, the framework is not equipped to deal with the load. The second is that, due to lack of necessity in this work, it does not facilitate the use of feature selection algorithms.

6.3 Future Work

This work leaves a few open issues to handle in the future.

As mentioned in the previous section, different data, due to different methods of collecting or processing it, may lead to worse results. This is an issue that is hard to deal with, as there are innumerous pieces of hardware that collect data in different ways and datasets. One possible solution is to review the features. Our work focused on statistical features due to their relative popularity, general decent results and extremely fast calculations. However, it may be possible to make good results consistent across datasets by using other features, though a good balance between number of complexity of features must be ensured in order to prevent the increase in estimation times. In addition to exploring different sets of features, adding different signals may also be a possible path to explore.

As also mentioned before, the framework is not prepared to handle very high volumes of data. Though it was not an issue through the development of this work, in future use it may be necessary to solve the problem.

A third point mentioned previously was the lack of proper feature selection functionality. Including this in the framework would allow the development of many emotion recognition systems that use these methods to reduce dimentionality.

As a possible future research topic, a study on using estimated values of Arousal and Valence to predict the quadrants or emotions as a possible improvement over the usual classification methods may be needed.

Bibliography

[1] Mohammed AbdelAal, Assem Alsawy, and Hesham Hefny. On emotion recognition using eeg. 12 2015.

[2] Juan Abdon Miranda-Correa, Mojtaba Khomami Abadi, Nicu Sebe, and Ioannis Patras. Amigos: A dataset for mood, personality and affect research on individuals and groups. *IEEE Transactions on Affective Computing*, PP, 02 2017.

[3] Jamilah Abdur-Rahim, Y Morales, Pankaj Gupta, Ichiro Umata, Atsushi Watanabe, Jani Even, Takayuki Suyama, and Shin Ishii. Multi-sensor based state prediction for personal mobility vehicles. *PLOS ONE*, 11, 10 2016.

[4] Adam Anderson, Thomas Hsiao, and Vangelis Metsis. Classification of Emotional Arousal During Multimedia Exposure. *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments - PETRA '17*, pages 181–184, 2017.

[5] Areej Babiker, Ibrahima Faye, Kristin Prehn, and Aamir Malik. Machine learning to differentiate between positive and negative emotions using pupil diameter. *Frontiers in Psychology*, 6(DEC):1–10, 2015.

[6] William Belson. Matching and prediction on the principle of biological classification. 8(2):65–75, 2013.

[7] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.

[8] Jason J Braithwaite, Derrick G Watson, Robert Jones, and Mickey Rowe. A guide for analysing electrodermal activity (eda) & skin conductance responses (scrs) for psychological experiments. *Psychophysiology*, 49(1):1017–1034, 2013.

[9] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 161–168, New York, NY, USA, 2006. ACM.

[10] Luca Chittaro and Riccardo Sioni. Affective computing vs. Affective placebo: Study of a biofeedback- controlled game for relaxation training. *International Journal of Human Computer Studies*, 72(8-9):663–673, 2014.

[11] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, January 1967.

[12] Priyanka Das, Anwesha Khasnobish, and D. N. Tibarewala. Emotion recognition employing ECG and GSR signals as markers of ANS. *Conference on Advances in Signal Processing, CASP 2016*, pages 37–42, 2016.

[13] Atefeh Goshvarpour, Ataollah Abbasi, and Ateke Goshvarpour. An accurate emotion recognition system using ECG and GSR signals and matching pursuit method. *Biomedical Journal*, 40(6):355–368, 2017.

[14] Atefeh Goshvarpour, Ataollah Abbasi, Ateke Goshvarpour, and Sabalan Daneshvar. Fusion Framework for Emotional Electrocardiogram and Galvanic Skin Response Recognition : Applying Wavelet Transform. *Iranian Journal of Medical Physics*, 13(3):163–173, 2016.

[15] Atefeh Goshvarpour, Ataollah Abbasi, Ateke Goshvarpour, and Sabalan Daneshvar. Discrimination between different emotional states based on the chaotic behavior of galvanic skin responses. *Signal, Image and Video Processing*, 11(7):1347–1355, 2017.

[16] Robert R. Henderson, Margaret M. Bradley, and Peter J. Lang. Emotional imagery and pupil diameter. *Psychophysiology*, 55(6):1–7, 2018.

[17] Yu Liang Hsu, Jeen Shing Wang, Wei Chun Chiang, and Chien Han Hung. Automatic ECG-Based Emotion Recognition in Music Listening. *IEEE Transactions on Affective Computing*, pages 1–16, 2017.

[18] S. Koelstra, C. Muhl, M. Soleymani, J. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, and I. Patras. Deap: A database for emotion analysis ;using physiological signals. *IEEE Transactions on Affective Computing*, 3(1):18–31, Jan 2012.

[19] Ann M Kring and Albert H Gordon. Sex Differences in Emotion : Expression , Experience , and Physiology. 74(3):686–703, 1998.

[20] T Bell Laboratories, Mountain Avenue, and Murray Hill. Random Decision Forests Tin Kam Ho Perceptron training. 1995.

[21] Mingyang Liu, Di Fan, Xiaohan Zhang, and Xiaopeng Gong. Human Emotion Recognition Based On Galvanic Skin Response signal Feature Selection and SVM. *International Conference on Smart City and Systems Engineering*, (November 2016):157–160, 2016.

[22] M. E. Maron. Automatic indexing: An experimental inquiry. *J. ACM*, 8(3):404–417, July 1961.

[23] Sadaf Moharreri, Nader Jafarnia Dabanloo, and Keivan Maghooli. Modeling the 2D space of emotions based on the poincare plot of heart rate variability signal. *Biocybernetics and Biomedical Engineering*, pages 1–16, 2018.

[24] Timo Partala and Veikko Surakka. Pupil size variation as an indication of affective processing. *International Journal of Human Computer Studies*, 59(1-2):185–198, 2003.

[25] Jonathan Posner and James A Russell. The circumplex model of affect : An integrative approach to affective neuroscience , cognitive development , and psychopathology. pages 715–734, 2017.

[26] Valorie N. Salimpoor, Mitchel Benovoy, Gregory Longo, Jeremy R. Cooperstock, and Robert J. Zatorre. The rewarding aspects of music listening are related to degree of emotional arousal. *PLOS ONE*, 4(10):1–14, 10 2009.

[27] L U Z Santamaria-granados, Mario Munoz-organero, Gustavo Ramirez-gonzález, Enas Abdulhay, and N Arunkumar. Using Deep Convolutional Neural Network for Emotion Detection on a Physiological Signals Dataset (AMIGOS). 2019.

[28] Jainendra Shukla, Miguel Barreda Angeles, Joan Oliver, G C Nandi, and Senior Member Ieee. Feature Extraction and Selection for Emotion Recognition from Electrodermal Activity. 3045(c), 2019.

[29] Paweł Tarnowski, Marcin Kołodziej, Andrzej Majkowski, and Remigiusz Jan Rak. Combined analysis of GSR and EEG signals for emotion recognition. *2018 International Interdisciplinary PhD Workshop, IIPhDW 2018*, pages 137–141, 2018.

[30] Kuan Tung, Po-kang Liu, Yu-chuan Chuang, Sheng-hui Wang, and An-yeu Andy Wu. Entropy-Assisted Multi-Modal Emotion Recognition Framework Based on Physiological Signals. *2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, pages 22–26, 2018.

[31] Sheng-hui Wang, Huai-ting Li, En-jui Chang, and An-yeu Andy Wu. Artificial Intelligence Applications and Innovations. 519(May), 2018.

[32] Wei Wei, Qingxuan Jia, Yongli Feng, and Gang Chen. Emotion Recognition Based on Weighted Fusion Strategy of Multichannel Physiological Signals. 2018(1), 2018.

[33] H. Yang and C. Lee. An attribute-invariant variational learning for emotion recognition using physiology. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1184–1188, May 2019.

[34] Roberto Zangróniz, Arturo Martínez-Rodrigo, José Manuel Pastor, María T. López, and Antonio Fernández-Caballero. Electrodermal activity sensor for classification of calm/distress condition. *Sensors (Switzerland)*, 17(10):1–14, 2017.

[35] Le-kai Zhang, Shou-qian Sun, Bai-xi Xing, Rui-ming Luo, and Ke-jun Zhang. Using psychophysiological measures to recognize personal music emotional experience. pages 1–12, 2018.